People's Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research
University of BATNA 2
Faculty of MATHEMATICS and COMPUTER SCIENCE

Thesis

Submitted for obtaining The 3rd cycle **Doctorat'S Degree**

In : Computer Science

Field of Study: Information Technology

By: Mr SEDDIK Mohamed Taki Eddine

Subject

Application of Machine Learning for Diagnosis in a Cloud Computing Context

Publicly defended, on 07/06/2023, in front of the jury composed of:

Pr.	MELKEMI Kamal Eddine	Prof.	at UB2	Presedent
Dr.	KADRI Ouahab	MCA	at UB2	Supervisor
Dr.	ABDESSEMED Mohamed Rida	MCA	at UB2	Co-Supervisor
Dr.	BOUCETTA Aldjia	MCA	at UB2	Examiner
Dr.	BOUSSAAD Leïla	MCA	at UB2	Examiner

Acknowledgment

In the name of Allah, the most gracious and the most merciful, I begin. My foremost and deepest gratitude is extended to Allah for His abundant blessings, guidance, and unwavering support that has bestowed upon me the strength, wisdom, and patience necessary throughout this journey. Every achievement and success in my life is a testament to His infinite grace.

I would like to extend my profound gratitude to my parents, Fartas Nouara and Seddik Ameur. Their ceaseless encouragement, unwavering faith, and enduring love have been the bedrock of my journey.

I extend my sincere gratitude to my family - my brothers and my aunts and uncles. Their unwavering support, constant encouragement, and steadfast belief in my abilities have been a source of strength throughout this journey.

My special appreciation goes to my advisor, Dr. Kadri Ouahab. His unwavering support, insightful guidance, and rigorous academic mentorship have been invaluable in shaping my intellectual and personal development throughout my doctoral studies. Similarly, I owe a profound debt of gratitude to my co-supervisor, Dr. ABDESSEMED Mohamed Rida. His astute feedback, consistent encouragement, and deep expertise have significantly enriched my research journey. The symbiotic guidance of both these esteemed scholars has been instrumental in bringing my thesis to fruition.

My gratitude is also due to my thesis committee members and my research group for their constructive feedback, encouragement, and for fostering an intellectually stimulating environment.

To my friends who have stood by me, shared laughs, and offered endless encouragement during this journey, I extend my heartfelt appreciation.

Finally,I am deeply thankful for the technical and administrative staff at my university. Their relentless service and dedication form the bedrock of our research environment.

Dedication

To my beloved parents, Fartas Nouara and Seddik Ameur,

This PhD thesis is dedicated to both of you, who have relentlessly instilled in me the principles of determination, hard work, and tenacity. Your unwavering faith in me, even in times when I doubted my own potential, has fortified me through countless trials and maintained my forward momentum. The wisdom you have passed on to me is mirrored in each and every page of this work. I am eternally grateful for your steadfast love, support, and faith, which have laid the solid groundwork for all of my achievements.

To my treasured grandparents, Fartas Boudjamaa and Mimoun Barkahoum,

whose memories continue to guide and inspire me. Even though you are no longer with us, your spirit and teachings continue to guide my steps. You live on through this work and every accomplishment I make.

To my mentors, my beacons of wisdom, Dr. Kadri Ouahab and Dr. ABDESSEMED Mohamed Rida.

Your insights have illuminated my path, guiding me through the intricacies of research and academic discovery. Your fervor for learning and your uncompromising commitment to scholarly excellence have molded me not only as an academic but also as an individual. Your mentorship has been instrumental in helping me chart the tumultuous waters of research and academia. The knowledge you imparted, your continuous encouragement, and your enduring patience during our intellectual exchanges have all contributed to the fruition of this odyssey. Like a lighthouse in stormy seas, you have guided me towards this monumental achievement.

To my brothers, Seddik Moussaab and Seddik Annes,

Your unwavering support and belief in me have always been a source of strength, spurring me on through the trials and tribulations of this journey. Your shared joy in my successes and your encouragement in moments of doubt have made this journey more bearable and fruitful.

To my aunts and uncles, especially Nabile,

Your unending love and guidance have played an instrumental role in shaping me, both personally and academically. Your life lessons, advice, and constant encouragement have been pillars of support throughout this process."

To my dear friends, especially Zatout Aymen, Hirech Hocine and Youkana Abderrahmane,

Your friendships have been a source of joy and comfort, your words of encouragement, a source of inspiration and courage to continue even in the toughest of times.

To my esteemed colleagues and fellow PhD students, particularly Bouhata Djamila, Younes Bouali, Mekadam Ayoub, Benbalgacem Mohamed, Bouafia Yassin, and Benbrahim Hayat,

Your companionship, camaraderie, and shared experiences have enriched not only this journey but also this work in profound ways.

This journey towards attaining my PhD has been a collaborative effort, with all of you acting as the sturdy pillars of support. This thesis is not merely a reflection of my work, but also a testament to your unwavering faith, patience, and love. Hence, it is with deep gratitude that I dedicate this thesis to all of you. Thank you.

الملخص

تتعمق هذه الرسالة في تطبيقات التعلم الآلي وقدرتها على التدخل في مجالين حاسمين؛ تعزيز الأمن في شبكات النقل البصري السريعة (OBS) ، ومواجهة التحديات المتعلقة بالبيانات المفقودة في الأنظمة الكهروضوئية (PV) استخدام التعلم الآلي ساهم في تقديم حلول منهجية للتحديات البارزة التي يواجهها كلا المجالين، وتكشف هذه الدراسة عن نتائج مهمة تحدد مسارات البحث المستقبلية في علم البيانات. يستعرض الجزء الأول من الدراسة ديناميكيات الأمن في شبكات (OBS) ، ويوضح كيف يمكن لنموذج التعلم الآلي الذي تم تدريبه بشكل صحيح أن يلعب دورًا حاسمًا في تعزيز قدرات الشبكة على مواجهة التهديدات. الجزء الثاني يركز على الأنظمة الكهروضوئية والتحديات المتعلقة بالبيانات المفقودة. نقدم منهجية جديدة تستخدم الانحدار بدعم الفيكتور (SVR) لتعبئة الفجوات في البيانات ونثبت فعاليتها. بشكل عام، النتائج المقدمة في هذه الرسالة تسلط الضوء على القدرة الثورية للتعلم الآلي على التعامل مع التعقيدات في الواقع. نظهر أن تأثير هذه التطبيقات يتجاوز بكثير الأمن الشبكي وإدارة الطاقة المتجددة، ويوحي بإمكانية وجود تطبيقات أكثر شمولاً في قطاعات أخرى.

الكلمات المفتاحية: التعلم الآلي، شبكات النقل البصري السريعة، الأمن الشبكي، الأنظمة الكهروضوئية، البيانات المفقودة، الانحدار بدعم الفيكتور

Abstract

This dissertation examines the transformative potential of machine learning applications in two critical but distinct domains: enhancing security in Optical Burst Switching (OBS) networks and addressing missing data issues in photovoltaic systems. Unfolding in two parts, the study systematically engages with each field's unique challenges through machine learning, revealing insightful outcomes that articulate the future trajectories of data science in real-world settings.

The first part navigates the security dynamics of OBS networks, showcasing how a properly trained machine learning model can play a pivotal role in bolstering these networks' resilience. The model significantly improves network performance and stability by accurately identifying and counteracting potential threats. This comprehensive exploration of OBS networks offers a promising solution for enhancing network security and uncovers the complexities that can emerge when implementing machine learning in these intricate systems.

The second part centers on photovoltaic systems and the perennial issue of missing data, a longstanding challenge in this field. We introduce an innovative methodology that employs support vector regression (SVR) to tackle these gaps and validate its efficacy. The results indicate that the SVR-based approach outperforms traditional data imputation methods in terms of accuracy and applicability, substantially enhancing the efficiency and reliability of photovoltaic systems. This part highlights machine learning's transformative ability to handle missing data and offers a deep understanding of the versatility of SVR applications.

In conclusion, the findings presented in this dissertation illuminate machine learning's transformative capacity to deal with real-world complexities. We demonstrate that these applications extend well beyond network security and renewable energy management, pointing to more pervasive applications in other sectors. Our research catalyzes future studies that delve deeper into machine learning, contributing to a more digitized future and technological advancement.

Keywords: Machine Learning, Optical Burst Switching (OBS) Networks, Network Security, Photovoltaic Systems, Missing Data, Support Vector Regression (SVR)

Résumé

Cette thèse examine le potentiel transformateur des applications d'apprentissage automatique dans deux domaines critiques mais distincts : améliorer la sécurité dans les réseaux de commutation optique en rafale (OBS) et aborder les problèmes de données manquantes dans les systèmes photovoltaïques. Dans ses deux parties, l'étude se penche systématiquement sur les défis uniques de chaque domaine par l'apprentissage automatique, révélant des résultats perspicaces qui articulent les trajectoires futures de la science des données dans les contextes réels.

La première partie navigue dans les dynamiques de sécurité des réseaux OBS, démontrant comment un modèle d'apprentissage automatique correctement formé peut jouer un rôle pivot dans le renforcement de la résilience de ces réseaux. En identifiant et en contrant avec précision les menaces potentielles, le modèle améliore considérablement les performances et la stabilité du réseau. Cette exploration complète des réseaux OBS offre non seulement une solution prometteuse pour améliorer la sécurité du réseau, mais dévoile également les complexités qui peuvent survenir lors de la mise en œuvre de l'apprentissage automatique dans ces systèmes intriqués.

La deuxième partie se concentre sur les systèmes photovoltaïques et le problème éternel des données manquantes, un défi de longue date dans ce domaine. Nous introduisons une méthodologie innovante qui emploie la régression à vecteurs de support (SVR) pour combler ces lacunes et valider son efficacité. Les résultats indiquent que l'approche basée sur le SVR surpasse les méthodes traditionnelles d'imputation de données en termes de précision et d'applicabilité, améliorant considérablement l'efficacité et la fiabilité des systèmes photovoltaïques.

En conclusion, les résultats présentés dans cette thèse illuminent la capacité transformatrice de l'apprentissage automatique à traiter les complexités du monde réel. Nous démontrons que ces applications vont bien au-delà de la simple sécurité du réseau et de la gestion de l'énergie renouvelable, pointant vers des applications plus répandues dans d'autres secteurs également. Nos recherches servent de catalyseur pour des études futures qui plongent plus profondément dans l'apprentissage automatique, contribuant à un avenir plus numérisé et à l'avancement technologique.

Mots Clée : Apprentissage automatique, Réseaux de commutation optique par rafales (OBS), Sécurité du réseau, Systèmes photovoltaïques, Données manquantes, Régression à vecteurs de support (SVR)

Contributions

International Article:

Takieddine Seddik, M., Kadri, O., Bouarouguene, C., & Brahimi, H. (2021). Detection of Flooding Attack on OBS Network Using Ant Colony Optimization and Machine Learning. *Computación y Sistemas*, 25(2), 423-433.

National Conference:

Seddik, M. T. E., Kadri, O., & Abdessemed, M. R. (2022, July). Imputation as Service Using Support Vector Regression: Application to a Photovoltaic System in Algeria. Paper presented at the 1st National Conference of Materials Sciences and Engineering (MSE'22), Khenchla, Algeria.

Contents

List of Figures

List of Tables

List of Acronyms

Chapter I

Introduction

Machine Learning (ML), diagnostics, and cloud computing are three interrelated fields that have witnessed significant growth in the past decade.

ML, a subset of Artificial Intelligence (AI), leverages statistical techniques to give computer systems the ability to "learn" from data without being explicitly programmed [1].

Diagnosis is crucial in healthcare; it helps identify diseases and conditions and guides treatment decisions. ML has the potential to improve diagnosis accuracy and speed by analyzing large amounts of data and identifying patterns that may not be visible to the human eye. Recent studies have shown promising results in using ML for diagnosis in various fields, such as dermatology, radiology, and cardiology. For example, a study published in the Journal of the American Academy of Dermatology found that a ML algorithm could accurately diagnose skin cancer with a sensitivity of a high percentage with different parameters [2]. Another study published in Nature Biomedical Engineering journal showed that a ML model could predict cardiovascular risk factors from retinal images with high accuracy [3].

Cloud computing refers to delivering various services through the Internet, including data storage, servers, databases, networking, and software [4]. With its massive computational power, cloud computing has become an essential component of ML infrastructure, especially for training complex models.

The intersection of these fields presents a promising opportunity. By leveraging cloud computing, ML models can be trained on large-scale datasets and then used for diagnostics. This setup can significantly enhance the speed and efficiency of diagnosing various issues, making it particularly beneficial for sectors like healthcare, manufacturing, and customer service [5, 6].

This study aims to delve deeper into the relationship between these three fields and explore potential improvements and breakthroughs that can be achieved.

I.1 Problem Statement and Motivation

The research addresses two significant and complex issues: safeguarding Optical Burst Switching (OBS) networks from Burst Header Packet (BHP) flooding attacks and managing missing data in Photovoltaic (PV) installations.

OBS networks represent an innovative development in optical communications. These networks, however, are susceptible to BHP flooding attacks, potentially resulting in severe denial of service scenarios that impair functionality and network service. Given the critical importance of OBS networks in today's telecommunication infrastructure, ensuring their robustness and security is vital. The motivation for investigating this issue arises from the urgent need to bolster network security, protecting critical communication channels from malicious cyber threats.

In the context of renewable energy, PV installations are pivotal in driving the shift toward sustainable energy sources. The efficient operation of these installations relies heavily on the availability of complete and accurate meteorological data. Missing or incomplete data poses a challenge, leading to potential performance inefficiencies and inaccuracies in energy generation predictions. The motivation behind addressing this problem stems from the growing global emphasis on renewable energy generation and the need to optimize the effectiveness of these installations.

I.2 Research Objectives and Questions

The principal objective of this research is to leverage Machine Learning (ML) techniques to resolve the identified problems in both OBS networks and PV systems.

Several research questions guide this investigation:

How can ML techniques be employed to detect and mitigate BHP flooding attacks in OBS networks? What ML models are most effective for this purpose, and what are their relative strengths and weaknesses? How can ML techniques manage missing data in PV installations, particularly Support Vector Regression (SVR)? How does an SVR-based imputation method compare with other imputation techniques regarding accuracy and efficiency? What strategies can be developed to improve data imputation techniques for PV installations further?

I.3 Research Significance

The potential impact of this research is substantial in several respects. By presenting robust and practical solutions to enhance the security of OBS networks, this study contributes to maintaining the integrity of essential telecommunication infrastructure. Similarly, by addressing the issue of missing data in PV systems, the research can significantly improve the operational efficiency of renewable energy installations.

The findings of this study have broad implications beyond these specific applications. The methodologies and strategies developed in this research can be adapted to other networks and renewable energy systems, expanding the scope of ML applications. Additionally, by highlighting the role of ML in tackling complex real-world issues, this research underscores the transformative potential of ML in various domains.

I.4 Scope and Limitations

The research is focused on OBS networks and PV installations, considering the unique characteristics and challenges associated with each. While the methodologies and findings may have implications for other network and energy systems, the ML models developed in this study specifically address the challenges presented in OBS networks and PV systems.

This study acknowledges several potential limitations. The complexity of real-world network scenarios may affect the generalizability of the security model developed for OBS networks. Similarly, the quality and completeness of the meteorological data used for PV system analysis may influence the efficacy of the proposed data imputation method. Furthermore, as with any ML-based approach, the success of the models is contingent on the quality, relevance, and representativeness of the input data. Discrepancies between the data used for model development and the real-world scenarios could limit the model's performance.

I.5 Thesis Structure

Chapter I: Introduction and Thesis Overview

This chapter introduces the central themes of the thesis, including ML, cloud computing, and their application in diagnostic systems. It provides a road map for the rest of the thesis.

Chapter II: Literature Review

This chapter delves into the existing literature on ML and cloud computing in diagnostic

systems. It provides a comparative analysis of ML algorithms used in diagnostics, explores the impact of cloud-based ML on these systems, and discusses the challenges presented by missing data. The chapter concludes by identifying gaps in existing literature and opportunities for future research.

Chapter III: Theoretical Framework

Here, the key theoretical concepts underpinning the thesis are explained. These include ML and its types and algorithms, missing data and its implications, and the role of cloud computing in ML. The chapter aims to comprehensively understand these concepts before presenting the methodology and application cases.

Chapter IV: Methodology

This chapter details the research design, data sources, preprocessing, and the implementation of ML models. It provides a rationale for the chosen design and model selection and describes the validation techniques employed.

Chapter V: Application Cases

This chapter contains two case studies. The first case study focuses on detecting flooding attacks on Optical Burst Switching (OBS) networks. The second case study deals with imputation using Support Vector Regression (SVR). Each case study presents a real-world application of previously discussed concepts and methodologies.

Chapter VI: Discussion

This chapter provides a critical analysis of the findings from the application cases. It links the theoretical framework and methodologies employed with the practical outcomes derived from the case studies.

Chapter VII: Conclusion

This chapter provides a comprehensive summary of the research, key findings, and implications for future research. It also offers final remarks, encapsulating the value and significance of the study.

Each of these chapters contributes to the overall goals of the thesis, weaving a narrative that illustrates the potential of ML and cloud computing in the context of diagnostic systems.

Chapter II

Literature Review

II.1 Introduction

In the relentless pursuit of scientific and technological progress, the intersection of ML and diagnostics has emerged as an up-and-coming area of research. The potential of ML to revolutionize diagnostic practices has led to a burgeoning body of literature dedicated to exploring and exploiting these prospects. Furthermore, the growing integration of cloud computing in this space presents an array of possibilities, both promising and complex. Despite the significant advancements, numerous challenges and gaps persist, posing obstacles to the optimal deployment of these technologies. This chapter explores these challenges, offering a comprehensive analysis of the current state of research in the field.

We delve into the intricate dimensions of ML in diagnostics, investigating its transformative potential and the complications it introduces. We grapple with the 'black box' nature of many ML models, which offer little to no insight into the logic of their predictions despite their accuracy. We also confront the issue of the lack of robust evaluation methodologies, highlighting the need for datasets that more accurately reflect the multi-faceted nature of real-world clinical scenarios.

Parallelly, we probe into integrating cloud computing with ML in diagnostics, elaborating on the massive advantages the cloud offers in terms of scalable and distributed data processing and storage. Nonetheless, we also illuminate the pressing concerns of data privacy and security in the cloud context, emphasizing the need for efficient and practical solutions.

II.2 Role of ML in Diagnostic Systems

ML has precipitated a significant transformation in diagnostic systems, touching diverse sectors such as healthcare, manufacturing, and cybersecurity. The power of ML models in pattern recognition and anomaly detection has particularly leveraged their roles in these complex systems.

In this section, we aim to delve into the multi-faceted roles and diverse techniques that ML embodies in diagnostic systems. We explore a variety of methodologies within ML, including supervised and unsupervised learning, Deep Learning (DL), and reinforcement learning. Each technique contributes unique advantages in interpreting and processing complex diagnostic data, leading to more accurate and reliable predictions.

While we highlight specific examples and cases, this section maintains a broader perspective, emphasizing the generalizable impact of ML across various diagnostic systems. In doing so, we underline the utility and significance of ML in navigating complex challenges, such as the handling of missing data, that permeate across sectors and diagnostic scenarios.

Overall, this section endeavors to provide a comprehensive and nuanced understanding of the expansive role of ML in diagnostic systems. Intertwining theoretical constructs with practical instances illuminates the significant and transformative potential of ML techniques within the landscape of diagnostic systems.

II.2.1 Evolution of ML in Diagnostic Systems

Over several decades, the influence of ML in advancing diagnostic systems across various sectors has significantly escalated, as the literature review indicates.

In healthcare, ML applications have heightened the precision and rapidity of diagnoses. A groundbreaking study by Esteva et al. [7] showcased a DL model's capability to classify skin cancer on par with dermatologists. Similarly, Gulshan et al. [8] deployed ML techniques to identify diabetic retinopathy and macular edema in retinal fundus photographs. Zhang et al. [9] proposed a Deep Convolutional Neural Network (DCNN)-based framework for image denoising, leveraging residual learning, which exemplifies the remarkable capability of DL algorithms in enhancing denoising performance in medical imaging applications.

ML's contribution to diagnostic systems extends to the manufacturing industry. Sipos et al. [10] employed ML to predict log-based system maintenance requirements, enabling preventive maintenance and minimizing system downtime.

The field of cybersecurity has also experienced ML's profound impact. Buczak and Guven [11] delineated the application of ML techniques in intrusion detection systems, demonstrating an enhanced classification of benign and malicious network traffic. Seddik

et al. [12] used ML, specifically Support Vector Machine (SVM), and cloud computing to detect Burst Header Packet (BHP) flooding attacks in OBS networks, highlighting the growing role of these technologies in network security.

One critical issue that ML has substantially addressed is managing missing data in diagnostic systems. Rubin [13] was among the pioneers to discuss statistical methods for missing data imputation. More recently, Che et al. [14] employed recurrent neural networks for imputing missing values in time-series medical data.

The review of past studies underlines the vast potential and adaptability of ML in augmenting diagnostic systems across sectors. As we venture forward, it is expected that ML's role will further diversify, particularly in addressing challenges like managing missing data.

II.2.2 Recent Advancements

In recent years, advancements in ML have revolutionized the field of diagnostic systems, enabling significant strides in diverse sectors. A focal point of these developments is the imputation of missing data, which has long posed challenges for biomedical datasets. Lim et al. [15] proposed a novel Variational Autoencoder (VAE) architecture, NIMIWAE, which accounts for both ignorable and non-ignorable patterns of missingness in input features at training time. This methodology outperformed existing approaches for unsupervised learning tasks and imputation accuracy through statistical simulation.

Medical image analysis is another significant advancement within the ML sphere. Liu et al. [16] reviewed the progress of DL research in this field, identifying some limitations when algorithms are derived from small-scale medical datasets. Meanwhile, Bakas et al. [17] engaged in the BRATS challenge and determined the optimal ML algorithms for brain tumor segmentation using Convolutional Neural Network (CNN), progression assessment, and overall survival prediction. Rajkumar et al. highlighted the substantial potential of ML in elevating diagnostic accuracy and patient prognosis [18].

Recent advancements have also been directed towards predictive maintenance, an area gaining significant importance in sectors like manufacturing and aviation. Çınar et al. [19] proposed a predictive model identifying the minimal number of distribution transformers prone to failure. Actual data implementation in Cauca Department, Colombia, showed that the model allows a saving of 13%

Advancements have been noted in the realm of cybersecurity as well. Strecker et al. [20] conducted a comparative analysis of three distinct Internet of Things (IoT) cybersecurity algorithms currently in use for malware and intrusion detection and found them to be effective for the current landscape of IoT cybersecurity.

ML's impact is being felt even in the field of biometric identification, with gait recognition emerging as a promising area. Moon et al. [21] developed an ensemble DL framework that used data from multi-modal sensors embedded in insoles to identify individuals based on their gait patterns.

In the medical realm, DL technology has offered fresh perspectives for endometrial cancer diagnostics. Fremond et al. [22] found that DL facilitates integrative analysis of multi-modal image and molecular datasets with clinical outcomes, offering new pathways for diagnostics.

These advancements indicate the exciting evolution of ML-based diagnostic systems, pointing towards a future of increased accuracy, speed, and capability in handling complex data-related issues.

II.2.3 Comparative Analysis of ML Algorithms for Diagnostics

Before the comparative study, it is vital to grasp the broad applications of ML models in diagnostic systems across sectors, as highlighted in our review.

II.2.3.1 Summary of Reviewed Studies

The extensive literature review underscores the remarkable and diverse applications of ML in diagnostic systems. The specific ML techniques employed, their fields of application, and the critical contributions of each study are succinctly encapsulated in Table II.1.

Table II.1: Application of Specific ML Techniques in Various Fields

Study	ML Technique	Field of Application	Key Contributions
Esteva et al. [7]	DNN	Healthcare (Skin Cancer)	Skin cancer classification
Gulshan et al. [8]	DNN	Healthcare (Ophthalmology)	Diabetic retinopathy and macular edema identification
Zhang et al. [9]	DCNN	Medical Imaging	Enhancement of image denoising
Sipos et al. [10]	Multi-Instance Learning	Predicting maintenance in log-based systems	scheduled maintenance
Buczak and Guven [11]	Neural Networks, ,Clustering(Kmeans..)..	Cybersecurity	Enhanced network traffic classification
Seddik et al. [12]	SVM	Cybersecurity	BHP flooding attack detection
Che et al. [14]	RNN	Various (Missing Data Imputation)	Methods for missing data imputation
Lim et al. [15]	VAE	Various (Missing Data Imputation)	Addressing patterns of missingness
Liu et al. [16], Bakas et al. [17]	CNN	Medical Imaging	DL progress and brain tumor segmentation
Çınar et al. [19]	Regression Techniques, Decision Trees, or Neural Networks	Manufacturing (Predictive Maintenance)	Identification of transformers prone to failure
Strecker et al. [20]	RF, SVM, and KNN	Cybersecurity and IoT	IoT cybersecurity algorithms comparison
Moon et al. [21]	Ensemble DL	Biometric Identification (Gait Recognition)	DL framework for gait recognition
Fremond et al. [22]	DL	Healthcare	Analysis of multimodal datasets

II.2.3.2 comparative study

A plethora of ML algorithms have been developed and applied in diagnostic systems, each with its unique strengths, limitations, and use cases. Understanding these algorithms' characteristics and their applications in diagnostics is crucial for developing effective and efficient diagnostic systems. Detailed information about each of these algorithms is provided below:

- **Logistic Regression** is a simple and interpretable algorithm often used for disease

risk prediction [23, 24].

- **Decision Trees** are easy to interpret and handle categorical data, making them suitable for disease classification [25, 26].
- **Random Forests**, which can handle complex relationships, are often used for predictive modeling in disease outbreaks [27, 28].
- The high performance of **Gradient Boosting** algorithms is advantageous for disease progression prediction [29, 30].
- **Support Vector Machines (SVMs)** , which handle high dimensional data, are typically utilized in imaging diagnostics [31, 32].
- **DL Neural Networks** are known for handling large and complex datasets, making them useful in genomics and electronic health records [33, 34].
- **K-Means Clustering** is efficient for large datasets and used for patient segmentation [35, 36].
- Lastly, **Principal Component Analysis (PCA)** reduces dimensionality, which is beneficial for genetic data visualization [37, 38].

Table II.2: Comparative analysis of ML algorithms for diagnostics.

Algorithm	**Strengths**	**Applications in Diagnostics**
Logistic Regression	Simple and interpretable [23]	Disease risk prediction [24]
Decision Trees	Easy to interpret, handles categorical data [25]	Disease classification [26]
Random Forest	Handles complex relationships [27]	Predictive modeling in disease outbreaks [28]
Gradient Boosting	High performance [29]	Disease progression prediction [30]
SVMs	Handles high dimensional data [31]	Imaging diagnostics [32]
DL	Handles large and complex datasets [33]	Genomics, electronic health records [34]
K-Means Clustering	Efficient for large datasets [35]	Patient segmentation [36]
PCA	Reduces dimensionality [37]	Genetic data visualization [38]

II.2.4 Key Contributions and Emerging Trends in ML for Diagnostic Systems

ML has made significant strides in multiple fields, including diagnostic systems, traffic safety research, education, human factors, photonic networks, and Urdu optical character recognition. These contributions illustrate the breadth and versatility of ML applications [15–17, 19–22, 39–43].

In diagnostic systems, ML has provided robust solutions to complex problems such as missing data imputation, medical image analysis, predictive maintenance, cybersecurity, and biometric identification. These advancements have set the stage for further exploration in these domains [15–17, 19–22].

ML's application in traffic safety research has resulted in reliable and accurate crash severity modeling. Future studies must address the persisting issue of imbalanced data and the reliability of evaluation metrics in ML models [39]. In education, ML techniques have been employed to predict student academic performance, offering actionable insights for students and educational institutions [40].

Human factors research has examined the unique challenges and contributions posed by ML, especially in the context of human-system interactions, design, evaluation, and training [41]. In photonic networks, ML has automated complex decision-making processes, enhancing various aspects of resource allocation, user data analysis, and service restoration [42]. ML has also revolutionized Urdu optical character recognition, introducing new possibilities for image acquisition, pre-processing, segmentation, feature extraction, classification/recognition, and post-processing [43].

Despite these impressive achievements, several challenges and future directions need attention. Ensuring the generalizability of ML models across different settings and populations remains a primary concern. Future research should focus on methodologies that can handle large-scale datasets more effectively and efficiently. Moreover, opportunities for fusing ML with other emerging technologies, such as IoT and 5G networks, should be exploited to create more integrated systems [44].

In conclusion, ML has significantly influenced various domains, setting a solid foundation for future research. The potential of ML to transform diagnostic systems and other fields is immense, making it a cornerstone of future technological advancements [45].

II.3 Use of Cloud Computing in ML

As we transition into an era dominated by digital technologies, the intersection of cloud computing and ML is transforming the landscape of various industries. Cloud computing

offers virtually unlimited storage and computational resources that facilitate the processing of big data and the execution of complex ML algorithms. Meanwhile, ML brings intelligence to the cloud, enabling it to deliver more sophisticated services that can understand, learn, predict, adapt, and potentially operate autonomously. This section explores the use of cloud computing in ML, focusing on its impact, frameworks for diagnostic systems, comparison with local deployment, and associated security and privacy concerns.

II.3.1 An Overview of Cloud Computing in the Context of ML

Cloud computing has revolutionized the way ML is implemented, especially with the advent of big data. As data volume, variety, and velocity increase, traditional on-premise infrastructures often fall short in terms of storage and computational capacity [46]. In this context, cloud computing provides scalable and elastic resources that can be provisioned according to demand.

More specifically, cloud computing offers Infrastructure as a Service (IaaS) , Platform as a Service (PaaS) , and Software as a Service (SaaS) [47]. IaaS delivers virtualized computing resources over the internet. This eliminates the need for investing in and maintaining physical computing infrastructures. PaaS offers an environment for developing, testing and managing applications. SaaS allows users to connect to and use cloud-based applications over the internet.

These services play a crucial role in the context of ML. IaaS enables the storage and processing of large volumes of data, PaaS facilitates the development and testing of ML algorithms, and SaaS offers ready-to-use ML applications [48]. This model also provides cost-effectiveness, as users pay only for the services they use. The scalability and flexibility of cloud services have democratized access to ML technologies, promoting innovation across various sectors.

II.3.2 Impact of Cloud-based ML on Diagnostic Systems

Implementing cloud-based ML has significantly impacted diagnostic systems in various domains, including healthcare, finance, and industrial processes [49].

In healthcare, for example, ML algorithms trained on cloud-based platforms can analyze a patient's medical records, compare them with vast health databases, and generate diagnostic suggestions [50]. This has the potential to reduce diagnostic errors and improve patient outcomes significantly. Similarly, in finance, ML algorithms can analyze a customer's financial transactions to detect anomalies that might indicate fraud. In industrial processes, ML algorithms can analyze sensor data to predict and prevent

equipment failures.

These examples demonstrate the transformative potential of cloud-based ML in diagnostics. The ability to access and analyze vast amounts of data, coupled with the cloud's computational power, opens up new possibilities for timely and accurate diagnostics [49].

II.3.3 Cloud-Based ML Frameworks for Diagnostic Systems

Numerous cloud-based ML frameworks have been developed to support the implementation of diagnostic systems. Among them, TensorFlow, PyTorch, and Apache MXNet are widely used [49].

TensorFlow, developed by Google, provides a comprehensive and flexible ecosystem of tools, libraries, and community resources that enables researchers to build and deploy ML applications quickly [51]. PyTorch, developed by Facebook's AI research group, is known for its simplicity and ease of use, making it a favorite among researchers for developing prototypes and experimenting with new ideas [52]. Apache MXNet, supported by Amazon, is recognized for its efficiency in training and deploying deep neural networks [53].

These frameworks have been optimized for distributed computing, enabling them to leverage the power of cloud computing. Moreover, cloud service providers like Amazon, Google, and Microsoft offer ML as a Service (MLaaS) platforms, providing pre-trained models, Auto ML capabilities, and other tools to simplify and accelerate the development and deployment of ML applications [48].

II.3.4 Evaluation of Cloud-based vs. Local ML Deployment in Diagnostics

Several factors come into play when comparing the efficacy of cloud-based and local deployments of ML in diagnostics.

Local deployment has advantages, particularly regarding data security and latency. Data processing locally doesn't have to be transmitted over a network, reducing the risk of interception or tampering [54]. This also eliminates network latency, allowing for faster processing and real-time diagnostics. However, the main drawback of local deployment is the significant investment required in hardware, software, and maintenance. Furthermore, local systems' computational and storage capacities are limited, restricting the scale of ML applications [55].

On the other hand, cloud-based ML offers scalability, flexibility, and cost-effectiveness. Cloud services can be scaled up or down according to demand, providing unlimited storage and computing resources. This enables processing larger datasets and more complex ML

models [47]. The pay-per-use model of cloud services also allows for more efficient resource utilization and reduced costs. However, cloud-based ML may be limited by network speed and pose security and privacy risks [54].

Therefore, a hybrid approach that combines the advantages of both local and cloud-based deployments may be the most effective solution. This approach can process sensitive data locally, while non-sensitive data and computationally intensive tasks can be offloaded to the cloud [47].

II.3.5 Security and Privacy Concerns in Cloud-based ML

While cloud-based ML offers many advantages, it also raises security and privacy concerns. Data confidentiality can be compromised if proper encryption methods are not employed. Furthermore, since data is often stored in shared environments in the cloud, there is a risk of unauthorized access or data leakage [54].

Moreover, privacy regulations such as the General Data Protection Regulation (GDPR) in the European Union and the Health Insurance Portability and Accountability Act (HIPAA) in the United States impose strict guidelines on how personal data can be stored and processed in the cloud [48]. Violation of these regulations can result in hefty fines and damage to reputation.

To address these challenges, cloud service providers must implement robust security measures such as encryption, identity and access management, intrusion detection systems, and regular security audits. In addition, they must ensure that their services comply with all relevant privacy laws. On the user side, it's essential to understand the security and privacy implications of using cloud services and to take necessary precautions, such as anonymizing data before uploading it to the cloud [4].

II.4 Challenges of Missing Data in Machine Learning

The performance and efficacy of ML models depend heavily on the quality and quantity of the data they are trained on. However, when dealing with real-world data, missing data is a common problem that researchers and practitioners face. This poses significant challenges, as missing data can compromise the quality of the models, leading to inaccuracies in predictions and, in extreme cases, rendering models ineffective. This section provides a comprehensive examination of the challenges of missing data in ML, the implications of these issues, effects on diagnostic accuracy, a variety of methods developed to tackle these challenges, and a discussion on the effectiveness of different data imputation techniques.

Finally, the section concludes with a look into the future of handling missing data, reviewing advanced approaches and emerging trends.

II.4.1 Understanding the Implications of Missing Data in ML

The absence of or missing data can have severe implications for ML models. There are three main types of missing data: missing completely at random (MCAR), missing at random (MAR), and missing not at random (MNAR). Each type impacts the analysis differently, and different techniques are required to handle them effectively [56]. If the data is MCAR, the probability of a missing data point is independent of observed and unobserved data, which is the ideal scenario. In the case of MAR, the missingness can be explained by other observed data. MNAR is the most problematic since the missingness depends on unobserved data, and it is difficult to ascertain if the data is MNAR. Thus, understanding the type of missingness is crucial for selecting appropriate strategies for handling missing data [57].

II.4.2 Effect of Missing Data on Diagnostic Accuracy

Missing data can have significant adverse effects on the accuracy and performance of diagnostic systems. These systems often utilize ML algorithms that require comprehensive and reliable data to create accurate predictive models. However, if the training data contains missing values, it can lead to biases and inaccuracies in the trained models, subsequently affecting diagnostic accuracy. This is particularly critical in healthcare, where a minor diagnostic inaccuracy can potentially lead to a wrong treatment plan, negatively impacting patient outcomes [58].

II.4.3 Statistical and ML Approaches to Handle Missing Data

Various statistical and ML methods have been developed to mitigate the challenges posed by missing data. Traditional statistical approaches include simple imputation techniques like mean or median imputation, where missing values are filled with the mean or median value of the observed values. Other strategies include deletion methods, where instances with missing values are removed from the dataset. However, these approaches often make strong assumptions about the data and can introduce bias [59].

Recent advances in ML have introduced more sophisticated techniques to handle missing data. These approaches often involve using ML algorithms to predict the missing values based on other attributes in the dataset. For instance, algorithms like k-nearest neighbors (KNN) and regression imputation can be used to predict missing values [60].

There are also more complex techniques, like Multiple imputation (MI), that generate multiple predictions for the missing values to account for the uncertainty of the imputations [61].

II.4.4 Evaluation of the Effectiveness of Different Data Imputation Techniques

Different imputation methods have varying degrees of effectiveness, and their suitability often depends on the specific dataset and the nature of the missing data. Simple imputation methods like mean imputation can be effective when the data is MCAR, but they can distort the underlying data distribution and introduce bias when it is not MCAR. Likewise, deletion methods can lead to substantial loss of information and decrease the statistical power of the analysis [57].

On the other hand, ML-based imputation methods can better handle complex patterns in the data, making them suitable for MAR and even MNAR data. However, these methods can be computationally intensive and may lead to overfitting when the proportion of missing data is high. In these cases, MI techniques, which create MI s for each missing value and then combine the results, can provide more reliable estimates by accounting for the uncertainty of the imputations [61].

II.4.5 The Frontier of Missing Data: Advanced Techniques and Emerging Trends

Advanced techniques are continually being developed to handle missing data, including methods based on DL and probabilistic graphical models. These methods can model complex, non-linear relationships in the data and effectively handle high-dimensional data with intricate dependencies among the variables [62].

In terms of the future, the field is moving towards a more nuanced understanding of the different types of missing data and the importance of choosing appropriate techniques to handle them. This includes recognizing that domain knowledge can play a crucial role in addressing the challenges of missing data. For instance, understanding the reasons behind the missingness and incorporating this information into the imputation process can lead to more accurate and reliable models [57].

II.5 Identifying Gaps in Existing Literature

One fundamental step in pursuing scientific knowledge and understanding is to identify the gaps in the current literature. These gaps often serve as the starting point for new research, paving the way for innovative solutions and advancements. In the context of ML in diagnostics and the use of cloud computing, along with the problem of missing data in ML, there are numerous aspects that the current literature has not sufficiently addressed. This section will systematically explore these unaddressed aspects, identify the shortcomings in the everyday use of technology, elucidate unresolved issues, and identify opportunities for future research and development.

II.5.1 Unaddressed Aspects in the Application of ML in Diagnostics

While ML has significantly transformed the field of diagnostics, there are certain aspects that the current literature does not fully address. One such aspect is the interpretability of ML models. Despite their ability to provide accurate diagnoses, many models, particularly Deep Learning DL models, often function as "black boxes," providing little to no insight into the reasoning behind their predictions [63].

Furthermore, many studies in the literature use benchmark datasets for evaluation, which may not accurately reflect real-world clinical conditions. For instance, the datasets may be highly balanced or lack the complex, multi-modal data encountered in clinical settings. As such, the robustness of these models in real-world applications remains an open question [64].

II.5.2 Shortcomings in Current Use of Cloud Computing in ML

The intersection of cloud computing and ML has been a game changer, allowing for scalable, distributed processing and data storage. However, the current literature reveals some significant gaps in this area. One critical gap is the challenge of privacy and security of data in the cloud. While some work is on secure computation and differential privacy, these solutions are not always efficient or practical in the context of ML applications [65].

Another shortfall lies in the area of resource management. ML tasks can be resource-intensive, requiring substantial computational power and memory. However, there is limited research on efficient resource management strategies that balance the computational demands of ML tasks with the finite resources of cloud systems [66].

II.5.3 Unresolved Issues Pertaining to Missing Data in ML

Missing data poses a significant challenge in ML, and while numerous methods have been developed to handle this problem, there are still unresolved issues. One key issue is that many imputation methods assume the data is MAR. However, this assumption does not hold in many real-world scenarios, where data can be MNAR [56].

Moreover, while there are advanced techniques for handling missing data, such as multiple imputations MI and DL -based methods, these techniques can be computationally intensive and complex to implement. The current literature lacks comprehensive and accessible guides on the practical implementation of these advanced techniques [62].

II.5.4 Opportunities for Future Research and Development

The gaps and unresolved issues identified above present numerous future research and development opportunities.

In the realm of ML in diagnostics, the research could focus on developing interpretable ML models and designing robust evaluation methodologies that accurately reflect real-world conditions.

In cloud computing, future work could develop adequate data privacy and security strategies and optimize resource management for ML tasks in the cloud.

Concerning missing data, research efforts could be directed toward developing practical and efficient methods for handling non-random missing data and providing comprehensive guides for implementing advanced missing data techniques.

By addressing these gaps, future research can drive significant advancements in the field of ML, particularly in its application to diagnostics and its integration with cloud computing.

II.6 Conclusion

II.6.1 Summary of Key Findings

The comprehensive examination of the literature in this chapter has underscored the promising potential and remaining challenges of applying ML to diagnostics, particularly in the context of cloud computing. It is clear that while ML and cloud computing offer powerful tools for enhancing the precision and efficiency of diagnostics, considerable hurdles remain to be overcome.

ML models, for all their accuracy, often lack interpretability, making their predictions

challenging to understand. This issue, combined with the discrepancy between the idealized datasets used in many studies and the complex realities of clinical scenarios, presents significant barriers to applying these models.

Cloud computing, on the other hand, while enabling the scalability and distribution of data processing and storage, raises critical concerns about data privacy and security. Given the demanding nature of ML tasks, the management of computational resources also stands as an unresolved issue.

II.6.2 Implications for Future Research

The gaps identified in the existing literature provide fertile ground for future research. There is a pressing need to develop ML accurate but also transparent and interpretable models. Developing robust evaluation methodologies designed to reflect the complexity of real-world clinical scenarios is also an urgent priority.

From the perspective of cloud computing, creating efficient strategies to secure data privacy and optimize resource management in ML applications is an exciting direction for future work. Additionally, there is a pressing need for more research into dealing with the issue of missing data, especially non-random missing data, and providing clear, actionable guides on how to implement advanced data imputation techniques.

II.6.3 Final Remarks

In conclusion, the fusion of ML with diagnostics, aided by cloud computing, stands at the frontier of a significant scientific leap. While the challenges identified are undoubtedly complex, they provide clear direction for the next wave of research and innovation. By addressing these challenges, the field can significantly improve healthcare outcomes globally. The synthesis of ML, diagnostics, and cloud computing offers a tantalizing glimpse into the future of healthcare that we must strive to make a reality.

Chapter III

Theoretical Framework

III.1 Introduction

Chapter III provides a meticulous inquiry into three cornerstone domains shaping the future of technological advancements: ML, the conundrum of Missing Data, and the paradigm-shifting field of Cloud Computing. This chapter aims to afford readers a robust comprehension of these multifaceted topics while dissecting their foundational theories, numerous classifications, functions, applications, challenges, and prospective implications.

We inaugurate the discussion with ML, a fundamental pillar of AI, explicating an array of its types: Supervised Learning, Unsupervised Learning, Semi-supervised Learning, and Reinforcement Learning. Moreover, we voyage across the terrain of diverse ML algorithms, underscoring their significance in fabricating intricate AI systems.

Subsequently, the dialogue transitions to Missing Data, surveying its patterns and mechanisms and expounding on various techniques for managing such instances, an omnipresent phenomenon in real-world data analysis tasks.

The terminal segment of this chapter is devoted to Cloud Computing, shedding light on its service models, characteristics, advantages, and potential obstacles. We elaborate on the symbiotic relationship between Cloud Computing and ML, touching upon the advent of MLaaS, scalability, agility, and the intricate issues of data security and privacy. We further exemplify cloud-based environments widely utilized in state-of-the-art data analysis and ML applications.

III.2 ML

In the theater of life, we constantly encounter new actors on the stage of our experiences. Consider, for example, an interaction with an unfamiliar individual, let's call him Alaoua.

After a brief discourse with Alaoua, we immediately begin to form initial impressions. We might discern that he's intelligent, articulate, and considerate. These judgments, however, are not conjured out of thin air. They are the results of our capacity to categorize, which is an evaluation procedure based on our prior encounters with people who are comparable in some way. We draw upon this well of knowledge, applying it to decipher the patterns in Alaoua's behavior. We use our cognitive abilities to identify, categorize, and anticipate based on a collective framework of prior encounters.

This intricate process has long been a source of fascination, particularly among scientists and engineers, who are perpetually engaged in efforts to make machines mimic the wonders of the human brain. Imagine a machine that can learn from its own experiences and expertise, recognise patterns in large amounts of data, and then organise that information into meaningful categories, much like how humans mentally sort individuals into groups. That way, the computer may learn from our experiences with novel people like Alaoua and apply what it learns to future situations.

The proposition might appear to be plucked straight out of a science fiction novel, yet it is not only conceivable but already realized through a discipline known as ML. Now, let's embark on the journey to understand ML, commencing with a precise definition of what it constitutes.

definition III.1. TM. Mitchel [67] defines ML as "A computer program is said to learn from experience E concerning some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.

III.2.1 Glossary [68]

- `Accuracy`: The rate of correct predictions a model makes. We can define it mathematically as:

$$\begin{aligned} \text{Accuracy} &= \frac{\text{Correct Prediction}}{\text{Correct Prediction} + \text{False Prediction}} \\ &= \frac{\text{True Positive} + \text{True Negative}}{\text{True Positive} + \text{True Negative} + \text{False Positive} + \text{False Negative}}. \end{aligned} \tag{III.1}$$

- `Categorical Data`: The values of the features are discrete.
- `Class`: One of the label's listed target values.
- `Dataset`: A set of examples.

- `Euclidean distance`: The distance between two points using the following formula:

$$d(p,q) = \sqrt{\sum_{i=1}^{n}(p_i - q_i)^2}, \tag{III.2}$$

  where p and q are two points in an n-dimensional space.

- `False Negative`: The model mispredicts a negative class.
- `False Positive`: The model mispredicts a positive class.
- `Feature`: A value that describes an individual.
- `Feature vector` [69]: All the values that describe an individual.
- `Individual`, `instance`, or `example`: Refers to one row of our dataset.
- `Inference`: Predicting new unlabeled examples using a trained model.
- `K-fold cross-validation`: A technique of validation that optimizes the use of testing data [70].
- `Label`: Target in classification.
- `Labeled Example`: The example with a feature vector and a label.
- `Learning algorithm`: An algorithm that uses a dataset to create a model.
- `Model`: The result of a learning algorithm that can classify or predict new samples.
- `Outliers`: Values far off from most other values.
- `Overfitting`: Developing a model equivalent to the training data. On new data, the model fails to generate reliable predictions.
- `Oversampling`: Duplicating minority-class instances from a class-imbalanced dataset to build a more balanced training set.
- `Parameter`: Our model aims to optimize the configuration variables, for example, weights in Neural Networks.
- `performance`: A metric that judges how good our model is.
- `Precision`: The proportion of being correct when predicting the positive class. That is:

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}}. \tag{III.3}$$

- `Target`: The output.
- `Testing set`: A portion of the dataset used to test our learning algorithm's performance.
- `Testing phase`: Testing or validating the model created by the training phase using the testing set.
- `Training set`: A portion of the dataset used to train our learning algorithm.
- `Training phase`: We train our model using the learning algorithm with the training set and look for the best model parameters.
- `True Negative`: The model predicts a negative class correctly.
- `True Positive`: The model predicts a positive class correctly.
- `Underfitting`: Building a model with low predictability.
- `Undersampling`: Deleting majority-class instances from a class-imbalanced dataset to build a more balanced training set.
- `Unlabeled Example`: The example with features without a label.

III.2.2 ML Types

The vast and diverse domain of ML has been meticulously classified into four primary categories, as depicted in Figure III.1: Supervised Learning, Semi-Supervised Learning, Unsupervised Learning, and Reinforcement Learning [69]. Each category signifies a distinct approach to learning from data, adopting unique methodologies, and serving varying purposes. We will highlight these categories as we journey through the forthcoming sections, providing a concise yet comprehensive overview of their conceptual underpinnings and practical applications.

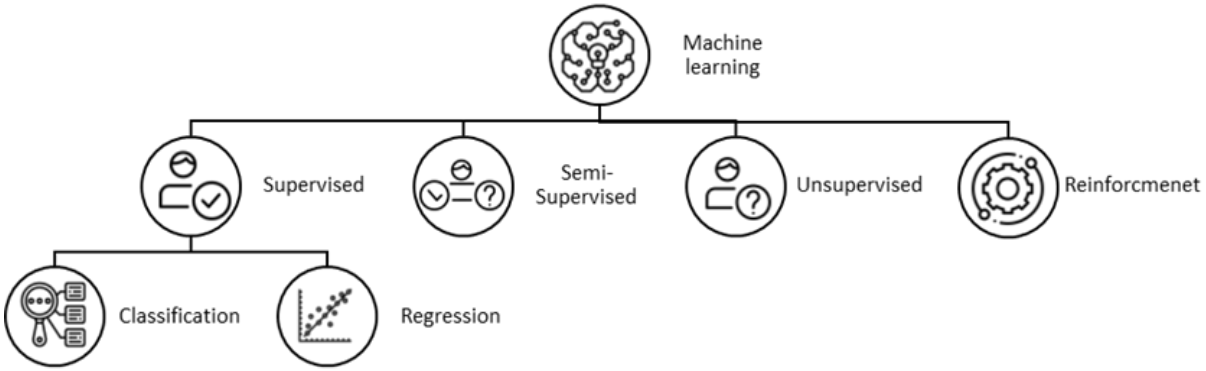


Figure III.1: ML Types

III.2.2.1 Supervised Learning

The cornerstone of our thesis is Supervised Learning, the most frequently employed paradigm in the realm of ML [71]. Its distinctive interaction with data sets supervised learning apart from its counterparts.

The nomenclature 'supervised' has its roots in the concept of guidance or supervision in deducing an output target from a given feature vector. In other words, it operates under the premise of learning from provided examples, each comprising a feature vector and an associated target. These targets, depending on the specific problem at hand, can either be categorical (class labels in a classification task) or continuous (real numbers in a regression task) [72].

A supervised learning algorithm leverages these annotated examples, inculcating in the machine an understanding of how to map the given feature vectors to their corresponding targets. Essentially, it equips the device with a model or function derived from the training data, which it can then apply to unseen instances. The machine can make accurate predictions or decisions when faced with new, unlabelled data by' learning' from this map of known feature-target pairings.

Supervised Model Creation Steps

The systematic process of crafting a supervised learning model (as outlined in Figure III.2) [69, 72, 73] consists of several deliberate steps. As seen below, these steps encapsulate the essence of supervised learning model development:

- **Selection of Dataset:** This forms the model's foundation. A dataset is a collection of examples denoted by E_i, where $0 < i < n$ and n is the total number of examples. Each example comprises a feature vector and a target, symbolized by $E_i = \{x_i, y_i\}$, with x_i representing the feature vector and y_i indicating the target. For instance, a

dataset consisting of patient records would be needed to construct a model discerning cancer patients. Each patient record (example) includes a feature vector with values signifying the patient's medical conditions, such as MRI scans, blood test results, etc., and a binary target—indicating if the patient's condition is malignant (true) or benign (false).

- **Algorithm Selection:** Choosing an optimal algorithm is paramount to the model's accuracy and precision.
- **Data Preparation:** Depending on the dataset and the ML technique employed, the data may require numerous preparation procedures, including noise reduction, missing data management, sorting, normalization, feature selection, and managing class-imbalanced datasets with undersampling or oversampling strategies, among others.
- **Data Splitting:** The dataset is partitioned into training and testing subsets (sometimes, a validation set is also created as a third subset). The proportion of this division depends on the dataset and the chosen ML algorithm.
- **Model Training:** The training set is utilized to instruct the learning algorithm and construct the model.
- **Model Testing:** The testing set is employed to evaluate the model's efficiency obtained from the training phase.
- **Model Deployment:** Once tested, the model (or classifier) is ready to be deployed for making predictions on unseen data.

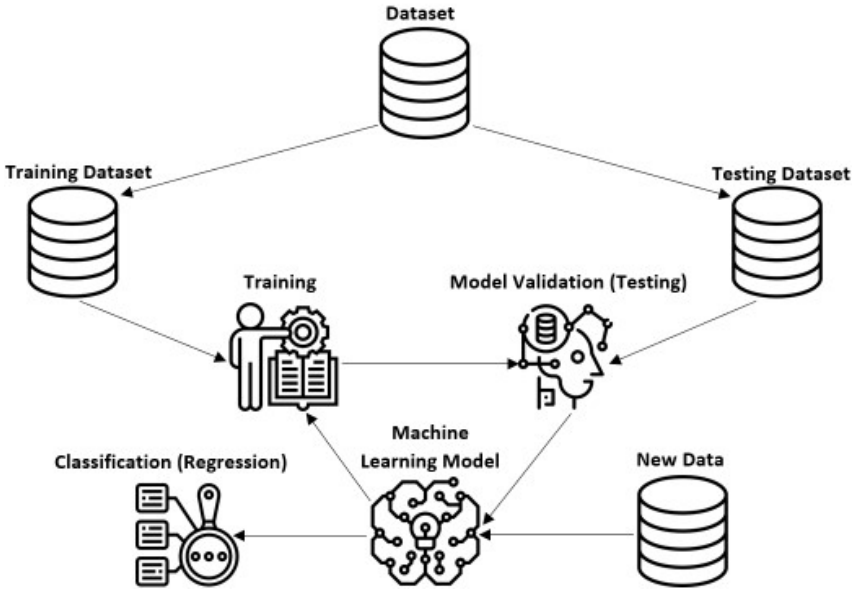


Figure III.2: Supervised ML Model

Supervised Learning categories

In the vast landscape of supervised learning, the type of output, whether it's categorical (discrete) or continuous (numerical), prompts us to differentiate between two primary categories [74]: Classification and Regression.

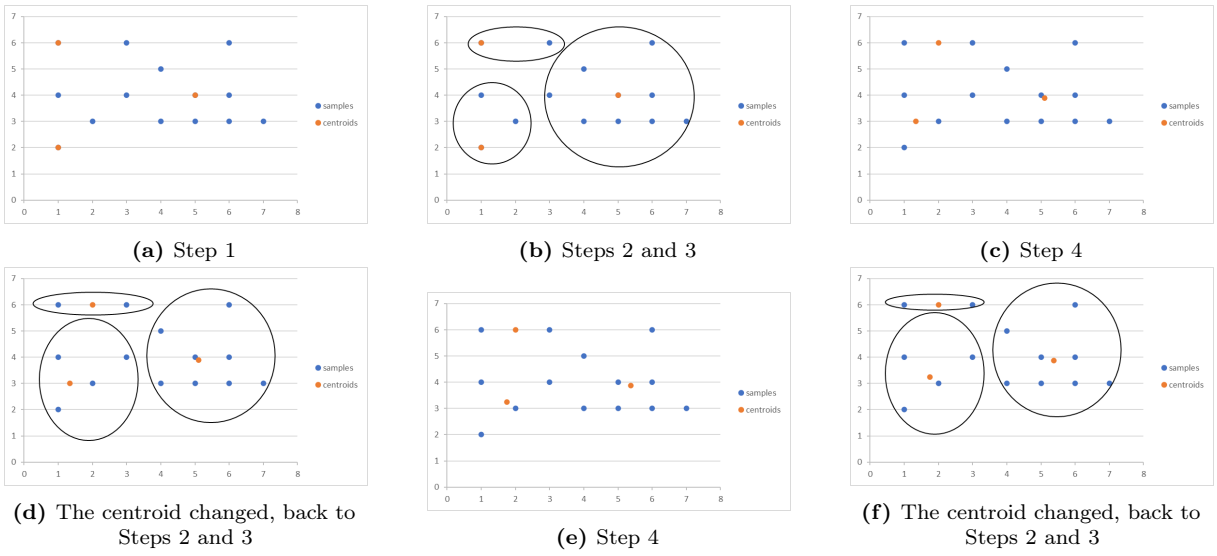
- **Classification:** A process that aims to assign categorical labels to incoming data instances. A classification algorithm aids in crafting a classifier that designates each new data instance to the corresponding class [69]. Examples of classification include diagnosing a patient's condition as benign or malignant, designating emails as spam or not spam, and identifying images as cat or dog. There are a variety of classification algorithms, such as SVMs, Decision Trees, k-Nearest Neighbor, and Neural Networks, which will be further scrutinized in upcoming sections.

- **Regression:** Unlike classification, regression pertains to instances where the output is a continuous or numerical value. A regression algorithm forms a model associating input features with a continuous output [69]. Practical instances of regression involve predicting a house's price based on its characteristics or estimating missing values in a dataset. Numerous regression algorithms exist, including Linear Regression, Neural Networks, and Decision Trees, which will be explored in greater depth in subsequent sections.

III.2.2.2 Unsupervised Learning

In the vast domain of ML, the ultimate aim of all algorithms is to construct a model. Unsupervised learning, however, stands distinct from supervised learning in that it does not rely on a 'teacher' providing an input-output mapping experience. In the unsupervised learning paradigm, the dataset comprises solely input data, or feature vectors, with no corresponding outputs [75]. Consequently, the model, distinct from supervised learning, navigates through the structure of these feature vectors, aiming to transform them into another vector or value, thereby extracting valuable insights [72].

Several approaches underlie unsupervised learning, including but not limited to clustering and dimensionality reduction. For instance, consider the K-means algorithm [76], a primary method in the clustering subfield. The K-means algorithm endeavors to segregate unlabeled data of size $n * f$ into K distinct clusters, where n denotes the number of examples and f symbolizes the number of features per example. The K-means algorithm unfolds in a series of steps, as delineated in Figure III.3:



(a) Step 1 **(b)** Steps 2 and 3 **(c)** Step 4

(d) The centroid changed, back to Steps 2 and 3 **(e)** Step 4 **(f)** The centroid changed, back to Steps 2 and 3

Figure III.3: K-means Algorithm Steps

- **Step 1: Initialization** - We commence by setting initial centroid values for each class. This can be performed randomly, or the first example in the dataset can be selected. Let us denote this as m_i, where $0 < i \leq K$ and $m_i = \{m_i^j / 0 < j < f\}$.
- **Step 2: Distance Calculation** - Subsequently, we calculate the Euclidean distance between all the examples in the dataset and the k established centroids.
- **Step 3: Assignment** - Each example in the dataset is assigned to the nearest centroid, thereby creating initial clusters.

- **Step 4: Centroid Update** - The centroids are recalculated using the mean of all examples currently assigned to each class.
- **Step 5: Convergence** - Steps 2 to 4 are iteratively repeated until the centroids do not experience any further change.

III.2.2.3 Semi-supervised Learning

Semi-supervised learning, as indicated by its terminology, functions as a strategic nexus between supervised and unsupervised learning methodologies [73, 77, 78]. In contrast to the typical supervised learning approach, semi-supervised learning employs a dataset where only a limited subset is labeled. Assigning labels to the unlabeled portion of the dataset often poses substantial challenges, owing to the intensive resource requirements, such as specialized equipment, expert human annotators, and considerable time investment [78].

The salient feature of semi-supervised learning lies in its remarkable adaptability. It can operate in both supervised and unsupervised modes, aligning its functionality with the dataset's inherent structure and the task's specific requirements. As illustrated in Figure III.4, a classifier can be trained using labeled examples and tested on unlabeled data in one scenario.

In another scenario, semi-supervised learning can leverage unlabeled data for tasks such as clustering. Here, estimates of cluster sizes can be derived based on the available labels. Moreover, when dealing with instances where some are labeled as 'q' and others as 'p,' constraints can be used to guide the learning process. For example, instances labeled as 'q' are linked together ('must-link'), whereas an instance labeled as 'p' is kept separate ('cannot-link') [78].

The flexibility of semi-supervised learning extends to domains beyond classification. It can be employed for regression tasks and dimensionality reduction and can accommodate a plethora of supervised and unsupervised algorithms. This versatility makes semi-supervised learning an attractive option for scenarios where labeled data are limited or expensive.

Nevertheless, caution is required when dealing with unlabeled data within unsupervised learning, as improper handling can lead to degradation in classification performance, negatively impacting the system's efficacy [79].

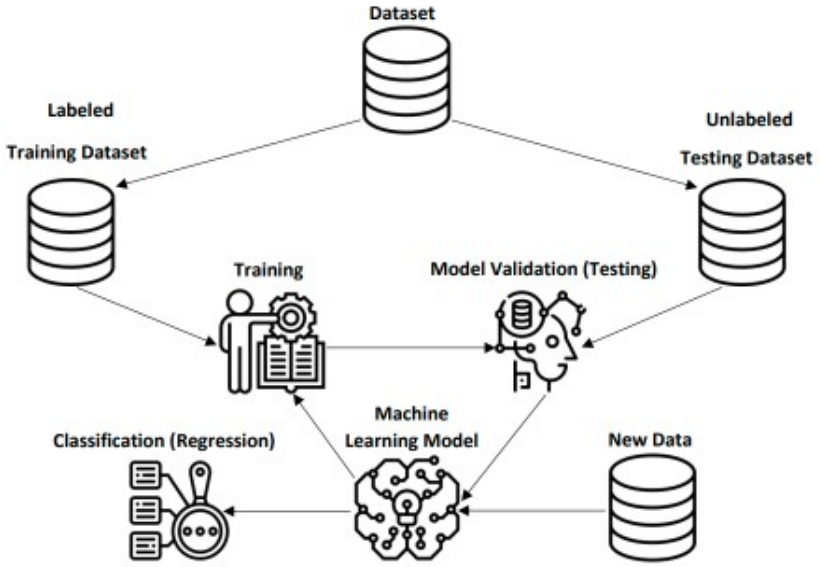


Figure III.4: Semi-Supervised Learning

III.2.2.4 Reinforcement Learning

Reinforcement learning (RL) offers a captivating perspective on learning analogous to observing a child learning to walk. The child, uncertain of the correct sequence of movements, engage in trial and error, gradually discerning which actions result in stability and progress [80]. Like this process, reinforcement learning deviates from the traditional supervised learning methods, which primarily depend on labeled datasets to predict or extrapolate outcomes. It also does not merely decipher the structure of unlabeled datasets like unsupervised learning. Instead, it frames learning as an optimization problem, where a sequence of actions is learned to maximize a long-term reward.

In the context of RL, the learner or decision-maker is termed an 'agent,' which interacts with its environment to achieve a goal. The agent navigates through a series of trial-and-error interactions or explorations without a clear guideline on what actions to execute. It gradually learns to select the activities that yield the most significant accumulated reward over time [80].

Figure III.5 illustrates the working of a reinforcement learning algorithm [81]. The interaction flow in RL unfolds as follows: The agent commences by acting within its environment. The system subsequently processes this action, which computes a 'reward' based on the action's effectiveness toward goal achievement. The computed reward, along with the new state of the environment post-action, is then relayed back to the agent. This cyclic action, reward, and feedback process form the crux of reinforcement learning.

This section offers a broad overview of reinforcement learning, refraining from delving into the intricacies of reward computation or goal specification. It serves as an introduction

to the fascinating RL paradigm, setting the stage for a more detailed exploration of its mathematical underpinnings, algorithms, and applications that follow.

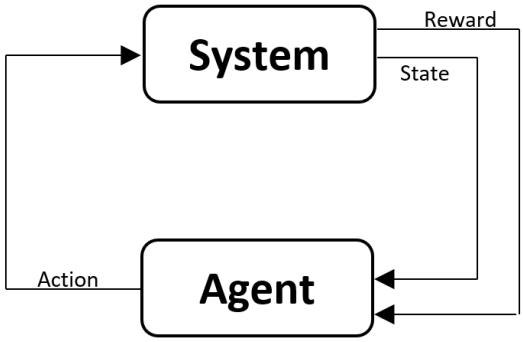


Figure III.5: Reinforcement Learning Algorithm

III.2.3 ML Algorithms

ML, a significant branch of AI, has demonstrated its importance across various scientific and industrial fields. This section predominantly focuses on supervised learning, a critical subfield of ML. Supervised learning models are generated based on known input-output pairs and are typically categorized into two main types: classification and regression. Although the nature of their output labels distinguishes these two types – categorical for classification and continuous for regression, it is noteworthy that the same underlying techniques can often be applied to both tasks.

Understanding the functionalities of these algorithms is crucial for comprehending the intricate workings of ML models, aiding in their effective selection, application, and evaluation within diverse problem contexts. Hence, we delve into a discussion on the vast array of supervised learning algorithms, highlighting their mechanics and unique features.

Upon closer inspection, one would observe that many of these algorithms are refined or modified versions of a few foundational ones [69]. This emphasizes the robustness and adaptability of these fundamental algorithms and manifests their pivotal role in shaping the evolution of supervised learning methodologies.

III.2.3.0.1 Linear Regression

Linear Regression often finds itself at the epicenter of a seemingly perennial debate: Does it belong to the realm of ML, or is it more aptly classified under statistics? According to Norman Matloff, a prominent figure in the Linear Regression discourse, the distinction

between a ML model and a non-ML model primarily hinges on the objective of prediction [82]. Indeed, Linear Regression has been recognized as a ML technique in numerous scholarly works [83, 84].

Let us delve into the working of Linear Regression without delving too deeply into technical intricacies (for a more thorough exploration, refer to [82]). As mentioned, we operate within a prediction model framework, aiming to predict a value y given a set of features x. Here, x serves as the input for our model, and y represents a continuous output value. The crux of our model is the optimal linear combination of the x features, represented as follows:

$$f_{w,b}(x) = b_0 + b_1 x \tag{III.4}$$

In this equation, b_0 represents the y-intercept, and b_1 denotes the slope. Our model training process involves the use of b_0 and b_1 as optimization parameters in the function $f_{w,b}(x_i) = b_{0,i} + b_{1,i}x_i = y_i$. Upon completion of the training, we end up with a model that provides the unknown y_p by combining the obtained b_0 and b_1 with the given x_p ($y_p = b_0 + b_1 x_p$). An example of a Linear Regression model is graphically demonstrated in Figure III.6, depicting the equation $Y = 0.3316X + 2.709$.

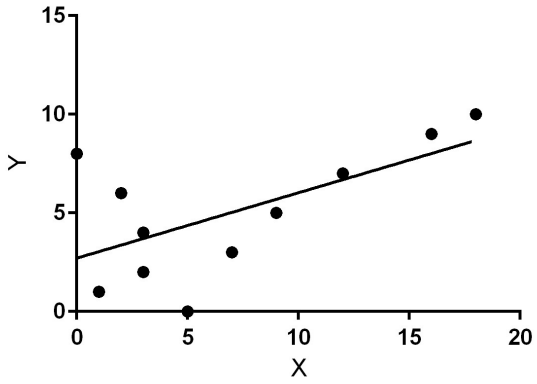


Figure III.6: Linear Regression example.

III.2.3.1 Probabilistic Classification

Probabilistic classification represents a class of methods that predict the class of a new instance by estimating its probability of belonging to all potential classes and then assigning it to the class with the highest probability. These methods compute the posterior class probability $P(y_i = k|x_i)$, where k ranges from 1 to C with C being the total number of possible classes [74].

Various algorithms emanate from the probabilistic classification framework, including but not limited to Naïve Bayes, Logistic Regression, and Probabilistic Graphical Models. However, these algorithms cannot be adequately understood or explained without a substantial background in probability theory.

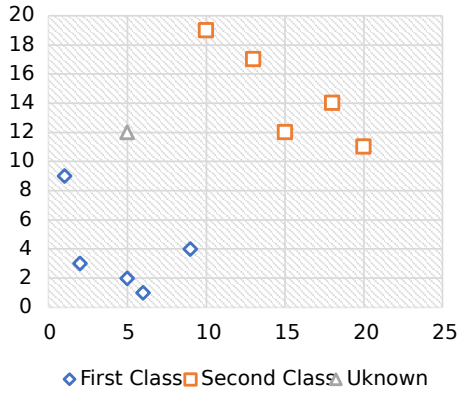
While probabilistic classification proves to be a powerful tool in the context of ML, it is important to note that this thesis will not be utilizing probabilistic classification methodologies. Consequently, we will refrain from delving into the specifics of these methods in this document. For those interested in gaining a more comprehensive understanding of probabilistic classification, the book "Data Classification: Algorithms and Applications" [74] is a recommended read.

III.2.3.2 KNN

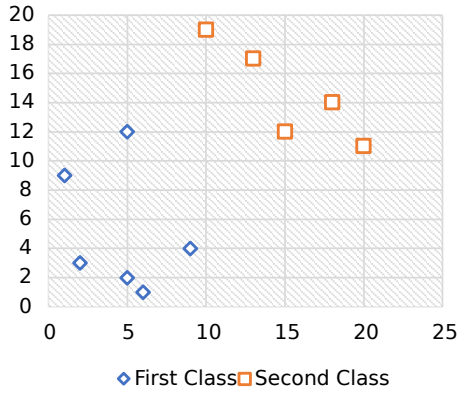
KNN algorithm was first proposed by Fix and Hodges in their study titled "Discriminatory Analysis, Nonparametric Estimation: Consistency Properties," published in The International Statistical Review (ISR), under project number 21-49-004, report number 4 [85]. This simple yet powerful algorithm has proven effective across various applications, often delivering satisfactory results.

As a supervised learning algorithm, KNN classifies new instances based on previous examples (instances with known classes). It operates through a straightforward series of steps. Initially, it computes the distance between the new instance and each of the existing instances. It then arranges these distances in ascending order, positioning the nearest instances at the forefront. The next step involves selecting the first 'k' instances from this sorted list, with 'k' representing a user-defined parameter that denotes the number of nearest neighbors to consider. Ultimately, the new instance is assigned to the most prevalent class among these 'k' instances.

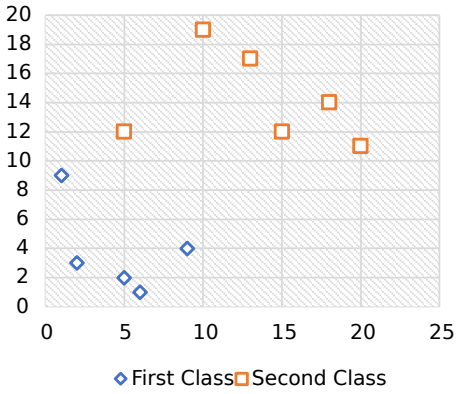
It is crucial to underscore the impact of 'k' on the classification outcome, as depicted in Figure III.7. A judicious selection of 'k' is paramount to ensuring the robustness of the KNN algorithm's performance.



(a) KNN model and unknown instance before classification.



(b) Classification result with $k = 1$.



(c) Classification result with $k = 3$.

Figure III.7: Example of KNN Classification.

In Figure III.7, subfigure III.7a illustrates the KNN model and unknown instance before classification, subfigure III.7b shows the result of applying classification with $k = 1$, and subfigure III.7c shows the result of applying classification with $k = 3$.

III.2.3.3 SVM

The SVM algorithm, since its inaugural introduction in the article "A Training Algorithm for Optimal Margin Classifiers" [86], has emerged as one of the most esteemed algorithms within the realm of ML [87]. This algorithm identifies a linear (or non-linear, employing kernels) hyperplane that best separates two classes. The distinguishing facet of the SVM algorithm is its pursuit of an optimal margin that segregates the support vectors of the classes, an aspect underscored in the title of the seminal article.

While our thesis incorporates SVM in the ensuing analysis, we shall provide an overview of the algorithm's operations without delving deep into the mathematical intricacies. It's also worth noting that SVM encompasses numerous variants, each tailored to specific prob-

lem domains. Nevertheless, we will concentrate on a two-dimensional, linearly separable case for conciseness and relevancy.

Figure III.8 portrays the SVM model post-training phase. The subsequent steps provide insights into how the algorithm reaches this optimal configuration:

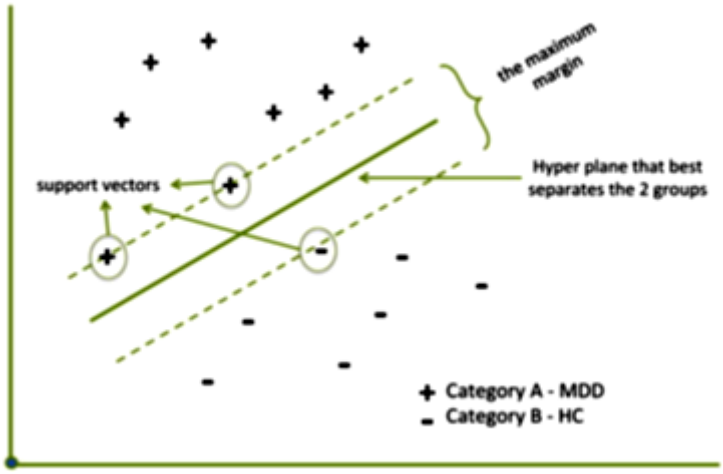


Figure III.8: SVM [88]

1. **Step 1:** Begin with a linearly separable dataset comprised of n instances, each marked by (x_i, y_i) where x_i is the set of m features, and y_i is the label corresponding to the instance i. For a visual representation, refer to Figure I-8. In the context of our variables:

   - i signifies the index of the instances with $0 < i < n$.
   - j designates the index of the features with $0 < j < m$.
   - x_i^j denotes the feature set for instance i.
   - y_i represents the label of instance i, defined as:

$$y_i = \begin{cases} 1, & \text{if } f(x_i) > 0 \\ -1, & \text{otherwise} \end{cases} \tag{III.5}$$

2. **Step 2:** We aim to identify a hyperplane (denoted as H_0 in Figure I-8) that effectively separates the two classes. The general equation for this hyperplane can be written as:

$$H_0 : \mathbf{w} \cdot \mathbf{x} + b = 0 \tag{III.6}$$

3. **Step 3:** Given the algorithm's objective to maximize the margin between the two classes and already have the separating hyperplane, we can derive two additional

hyperplanes (support vectors) based on the abovementioned equations. They are formulated as follows:

$$H_1 : \mathbf{w} \cdot \mathbf{x_i} + b \geq 1, \text{ if } y_i = 1 \tag{III.7}$$

$$H_2 : \mathbf{w} \cdot \mathbf{x_i} + b \leq -1, \text{ if } y_i = -1 \tag{III.8}$$

To consolidate the conditions from Equations III.7 and III.8, we get:

$$y_i(\mathbf{w} \cdot \mathbf{x_i} + b) \geq 1, \forall i \in \{1, \ldots, n\} \tag{III.9}$$

III.2.3.4 Decision Tree

Decision Trees are a fundamental instrument in the toolbox of ML practitioners, used across many classification and regression tasks [89]. They operate by successively partitioning the data space into distinct regions, each characterized by a different class or outcome. The resulting model is often intuitive, offering clear interpretability as it mimics human decision-making processes [90].

Although Decision Trees can handle both numerical and categorical data, they can potentially suffer from overfitting, where the model might perform exceptionally well on the training data but poorly on unseen data [91]. This overfitting problem can be mitigated through pruning, reducing the complexity of the final model [92].

In ML, numerous variants of Decision Trees exist, each with strengths and weaknesses. These variants include but are not limited to ID3, C4.5, CART (Classification and Regression Trees), and Random Forests [93].

A fundamental concept in Decision Trees is the criterion to decide on the best attribute or feature to split upon at each node. Based on Entropy, Information Gain is one commonly used criterion [90]. The aim is to choose the feature that provides the most homogenous child nodes, thereby reducing the uncertainty or Entropy.

Given a set of examples S, with a binary classification, the Entropy is given by:

$$H(S) = -p_+ \log_2(p_+) - p_- \log_2(p_-) \tag{III.10}$$

where p_+ is the proportion of positive examples in S and p_- is the proportion of negative examples in S.

The information gained from a feature A is then defined as the reduction in Entropy as follows:

$$IG(S, A) = H(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} H(S_v) \tag{III.11}$$

where $Values(A)$ is the set of all possible values for attribute A, and S_v is the subset of S for which attribute A has value v.

By using such criteria, Decision Trees continually partition the data into subsets, creating a tree-like decision model.

III.2.3.5 Artificial Neural Networks (ANNs)

Artificial Neural Networks (ANNs) are computing systems influenced by the neural networks that constitute animal brains, thereby enabling them to "learn" from observational data [94]. They are designed to replicate how the human brain functions, allowing the machine to learn from experience.

The basic building block of every ANN is an artificial neuron, or node, analogous to biological neurons in the brain. These nodes are connected by 'edges' which transmit data, with weights signifying the strength of these connections.

An artificial neuron j receives inputs from previous neurons, each multiplied by a corresponding weight w_{ij}. The neuron then computes the weighted sum of the inputs and optionally adds a bias term b_j, before applying an activation function f to this sum to produce the output y_j:

$$y_j = f\left(\sum_i w_{ij} x_i + b_j\right) \tag{III.12}$$

Each neural network comprises multiple layers, including an input layer, one or more hidden layers, and an output layer [95]. The layers between the input and output layers are known as 'hidden' because they are not visible externally. The nodes in these layers apply transformation functions, often non-linear, to their inputs before passing the results to the next layer.

The learning process involves iteratively adjusting the weights and biases of the network to minimize the difference between the actual and predicted outputs for a given set of input data—a process typically known as backpropagation [96].

The structure and behavior of ANNs make them exceptionally capable in various tasks, including image recognition, speech recognition, natural language processing, and anomaly detection, among others [97]. Moreover, the advent of DL, which involves neural networks with many hidden layers, has further propelled the application of ANNs to complex ML tasks.

Despite their notable performance in many areas, ANNs have some limitations. The 'black box' nature of these networks makes them quite difficult to interpret, which can be a significant problem in applications requiring transparency and accountability. Moreover,

training large and complex ANNs requires considerable computational resources and expertise in model tuning [98].

III.3 Missing Data

In the discipline of data science and ML, addressing the issue of missing data forms a crucial part of the overall analytical process. Missing data pertains to the absence of certain entries in the dataset that would otherwise contribute significantly to the study if they were accessible. This absence of data can potentially induce bias, limit the statistical validity of the study, and present complexities in the accurate interpretation of analytical outcomes. As such, gaining a thorough understanding of the causes and patterns of missing data and strategies to navigate these challenges is paramount in ensuring the rigor and veracity of any data-centric study. In the ensuing sections, we shall delve into the intricacies of these aspects of missing data and explore various methodologies to tackle them efficiently.

definition III.2. Little and Rubin articulate the concept of missing data as values that remain unobserved yet would hold considerable relevance to the study if they were indeed available for analysis [59].

III.3.1 Missing Data Patterns

The pattern of missing data refers to the arrangement or distribution of missing values within a dataset. Although numerous possible configurations exist of missing data, we concentrate on those most frequently appearing in the literature shown in Figure III.9, as outlined by Craig K. et al. [99].

Let us denote our dataset as a matrix $Y = (y_{i,j})$, where $i \leq n$ and $j \leq p$ (n is the number of rows corresponding to records or individuals, and p the number of columns or variables). We can define an indicator matrix $M = (m_{i,j})$ for the missing values such that $m_{i,j} = 0$ when $y_{i,j}$ is observed and $m_{i,j} = 1$ when $y_{i,j}$ is missing, as demonstrated in Table III.1. In the following examples, each consisting of eight records and four variables, we present diverse patterns of missingness, as depicted in Figure III.9. The missing values are visualized using the Python package 'missingno,' developed by Aleksey Bilogur [100].

Panel A of Figure III.9 illustrates a univariate pattern, signifying that all missing values are concentrated within a single variable, as seen with the fourth variable, Y_4, in our example. Panel B depicts a unit nonresponse pattern (or multivariate pattern), typically observed in surveys where certain participants decline to respond. The missing values are

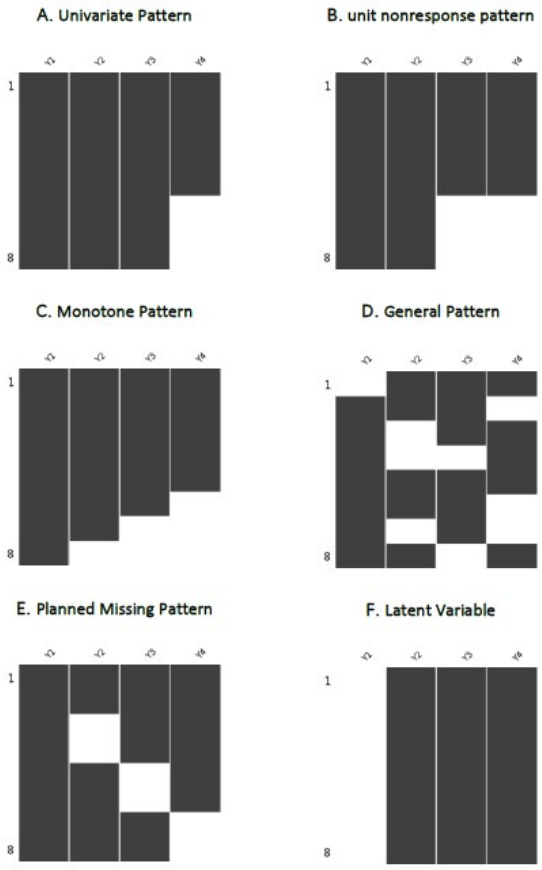


Figure III.9: Missingness patterns. The observed values are dark areas, and the blank areas show missing values.

found in the third and fourth variables, Y_3 and Y_4. The monotone pattern demonstrated in Panel C usually emerges when study participants prematurely cease participation, resulting in the absence of all following variables Y_k for a particular individual, where $j < k \leq n$.

Panel D shows the general pattern, arguably the most common type of missingness. This represents a random distribution of missing values across the dataset. The planned missing pattern in Panel E, as used by Graham et al., arises from implementing a three-form-based questionnaire where the form varies among different groups of individuals [101]. One group receives the complete form, while others receive forms with intentionally omitted questions, leading to a distinctive missing variable Y_j for each group. Lastly, the latent pattern is characterized by a missing variable for all individuals.

While various strategies in the literature address one or more of these patterns, maximum likelihood estimation, and MI s are the most comprehensive, dealing effectively with most patterns. Our discussion will explore how these patterns influence imputation and classification rates.

Table III.1: Missing values indicator matrix $M_{i,j}$

	A				B				C			
j\i	M1	M2	M3	M4	M1	M2	M3	M4	M1	M2	M3	M4
M1	0	0	0	0	0	0	0	0	0	0	0	0
M2	0	0	0	0	0	0	0	0	0	0	0	0
M3	0	0	0	0	0	0	0	0	0	0	0	0
M4	0	0	0	0	0	0	0	0	0	0	0	0
M5	0	0	0	0	0	0	0	0	0	0	0	0
M6	0	0	0	1	0	0	1	1	0	0	0	1
M7	0	0	0	1	0	0	1	1	0	0	1	1
M8	0	0	0	1	0	0	1	1	0	1	1	1
	D				E				F			
M1	1	0	0	0	0	0	0	0	1	0	0	0
M2	0	1	0	1	0	0	0	0	1	0	0	0
M3	0	1	0	0	0	1	0	0	1	0	0	0
M4	0	0	1	0	0	1	0	0	1	0	0	0
M5	0	0	0	0	0	0	1	0	1	0	0	0
M6	0	0	0	1	0	0	1	0	1	0	0	0
M7	0	1	0	1	0	0	0	1	1	0	0	0
M8	0	0	1	0	0	0	0	1	1	0	0	0

Table II-1: Example of six (A-F) missingness pattern matrix M used in Figure III.9. The rows and columns represent m_i and m_j, respectively. A missing value is denoted as 1.

III.3.2 Missing Data Mechanisms

III.3.3 Mechanisms of Missing Data

Rubin Donald [102] proposed a typology to classify the mechanisms of missing data into three distinct categories, each reflective of a unique correlation (or lack thereof) between the missing and observed variables in a dataset. As in the previous section, we denote the dataset matrix as $Y = (y_{i,j})$, partitioned into observed values Y_{obs} and missing values Y_{mis}, and the missingness matrix indicator as $M = (m_{i,j})$. We also introduce Φ, representing unspecified parameters that may impact missingness.

The missing data mechanism is defined as the probability distribution of the missingness indicator M. The dependency interplay among M, Y_{obs}, Y_{mis}, and Φ is crucial in distinguishing one mechanism from another, as illustrated in Fig II-2 [99].

To delineate the differences between these mechanisms, we employ a hypothetical survey conducted among two groups of university students. The first group is queried about their name, gender, age, preferred snack, and weight, while the second group is asked about their name, age, preferred snack, and weight.

- **MAR** : This mechanism implies that the probability of a variable $Y_{i,j}$ being missing is contingent on some observed variable $Y_{i,k}$ $(j \neq k)$, not on the missing variable $Y_{i,j}$ itself (Fig II-2, Panel A). For instance, if the variables age and weight are missing for female participants in the first group, the MAR mechanism would be in play. The missingness probability is related to the observed variable, gender.

$$P(M|Y) = P(M|Y_{\text{obs}}, \Phi) \tag{III.13}$$

  Equation II.1 signifies that the probability of missingness $P(M|Y)$ is associated with the unknown parameters Φ and the observed variables Y_{obs}.

- **MNAR or Not Missing At Random (NMAR)**: This mechanism dictates that missingness depends on some missing variables Y_{mis} and/or observed variables Y_{obs} (Fig II-2, Panel B). An instance of this would be missing weight values in the second group, indicating that missingness probability depends on the weight itself.

$$P(M|Y) = P(M|Y_{\text{mis}}, Y_{\text{obs}}, \Phi) \tag{III.14}$$

  Equation II.2 underscores the strong reliance on missingness probability on all dataset variables in the MNAR mechanism.

- **MCAR** : Under this mechanism, there exists no correlation between the missingness probability and any data, whether observed Y_{obs} or missing Y_{mis} (Fig II-2, Panel C). An example would be if the preferred snack field was missing in both groups, suggesting that the missingness occurred randomly, unconnected to any observed or missing variable.

$$P(M|Y) = P(M|\Phi) \tag{III.15}$$

III.3.4 Handling Missing Data

Handling missing data is essential in pre-processing for ML and statistical analysis. When no data value is stored for a particular observation in a variable, we say the data is missing. The types of missing data include MCAR, MAR, and Not Missing At Random (NMAR), which each have different impacts on the validity and power of subsequent analyses [103].

III.3.4.1 Classical Techniques

- **Listwise Deletion:** Listwise deletion, also known as complete case analysis, is one of the simplest methods to handle missing data. If any variable in an observation is

missing, listwise deletion will remove the entire observation from the dataset [104]. Although simple, this approach can significantly reduce the sample size and lead to biased estimates if the missingness is not completely random. Furthermore, it ignores the information in the non-missing elements of a partially missing observation, potentially wasting valuable information.

- **Pairwise Deletion:** Unlike listwise deletion, pairwise deletion, or available case analysis, does not exclude the entire observation when missing data is encountered. Instead, it excludes the specific missing value in the statistical analysis [105]. This method keeps as much data as possible and can be beneficial when the missingness is scattered randomly through the data. However, it can result in different subsets of data being used for different analyses, which may complicate the interpretation and comparison of results.

- **Mean Substitution:** Mean substitution is another straightforward method for handling missing data, where missing values are replaced by the mean of available cases for that variable [56]. While this method maintains the overall mean of the data, it artificially reduces the variance, covariances, and correlations, leading to potentially biased statistical inferences. It assumes that the data is MCAR and that the missing values are most likely close to the mean, which may not always be valid.

- **MI :** MI is an advanced statistical technique for handling missing data. Unlike previous methods, MI estimates the missing values multiple times to reflect the uncertainty around the unobserved data. The process involves three stages: imputation, where missing values are filled in to create multiple complete datasets; analysis, where each dataset is analyzed separately; and pooling, where the results from each dataset are combined to produce one overall result [103]. The strength of MI is that it provides unbiased parameter estimates and corrects standard errors under the assumption that the data is MAR.

III.3.4.2 ML Techniques

- **KNN Imputation:** KNN imputation method is a ML technique that fills missing values using the values of the k most similar observations. The "similarity" is usually defined by a distance metric, such as Euclidean distance. The advantage of KNN is that it can adapt to the local structure of the data. It works for both categorical and continuous variables and doesn't require an assumption about the distribution of data [106].

- **Random Forest Imputation:** The Random Forest method is a non-parametric algorithm that can be used for missing data imputation. It estimates missing values by creating multiple decision trees and averaging the predictions. It's a flexible method as it can handle non-linear relationships and interactions and is not influenced by outliers [107]. The random forest method does not assume that the data is MAR and can handle both numerical and categorical variables.

- **DL Methods:** DL methods like Autoencoders are recently being applied for missing data imputation. An autoencoder is a type of ANNs used for learning efficient data codings. An autoencoder aims to learn a representation for a data set, typically for dimensionality reduction. For missing data, we can train an autoencoder on the observed data and use it to generate the missing data by minimizing the reconstruction loss [108]. The advantage of autoencoders is their ability to capture non-linear relationships in the data, which can provide more accurate imputations than linear methods.

- **SVM Imputation:** SVM (SVM) imputation utilizes the capabilities of SVR to predict missing values within a dataset. The concept behind SVM imputation is to consider each feature with missing values as a dependent variable and all other features as independent variables. An SVR model is then trained to predict the missing values [109].

  SVR operates similarly to SVM . However, instead of finding a maximum margin hyperplane that separates classes, SVR seeks a function that approximates data points with maximum margin. SVR commonly uses an ϵ-insensitive loss function, which disregards errors that fall below ϵ. Like SVM, SVR can use the "kernel trick" to handle non-linear relationships between variables.

  The SVM imputation process follows these steps:

  1. For each feature with missing values, split the observations into two groups: one with observed values (training set) and one with missing values (test set).
  2. Use the observations with known values to train an SVR model, using the other features without missing values as predictors.
  3. Apply the trained SVR model to predict the missing values in the test set.
  4. Substitute the missing values with the predicted ones.

  SVM imputation, like other imputation methods, assumes a certain relationship exists between the feature with missing values and other features and that this

relationship can be captured by an SVM model. However, SVM imputation may be computationally costly if the dataset is large or has many features with missing values.

III.4 Cloud Computing

Cloud computing, a major shift in information technology, offers on-demand delivery of compute power, database storage, applications, and other IT resources via the internet with pay-as-you-go pricing [4]. This shift has revolutionized the computing world by liberating organizations from the need to build and maintain computing infrastructures in-house.

definition III.3. Cloud computing is a significant advancement in delivering information technology and services. By providing on-demand access to a shared pool of computing resources, everything from applications to data centers, over the internet, cloud computing enables organizations to avoid upfront infrastructure costs and focus on projects that differentiate their businesses instead of on infrastructure [110].

Moreover, cloud computing enables IT to rapidly implement innovative solutions to help businesses respond to market changes. However, it also poses significant potential risks and challenges in areas such as security, compliance, governance, and long-term costs [111].

III.4.1 Cloud Service Models

There are three primary cloud service models, which each represent different parts of the broader cloud computing architecture: IaaS, PaaS, and SaaS [4].

- **IaaS:** This model offers the infrastructure such as virtual machines and other resources like a virtual-machine disk image library, block and file-based storage, firewalls, load balancers, IP addresses, and virtual local area networks. Examples include Amazon Web Services (AWS), Microsoft Azure, and Google Cloud Platform.
- **PaaS :** This model is used for applications and other development while providing cloud components to software. Examples include Google App Engine and AWS Elastic Beanstalk.
- **SaaS :** In this service model, users gain access to application software and databases. Cloud providers manage the infrastructure and platforms on which the applications run. Examples include Google Apps, Microsoft Office 365, and Salesforce.

III.4.2 Characteristics and Advantages of Cloud Computing

Cloud computing services have several unique attributes and advantages [46], such as:

- **On-demand self-service:** Cloud computing resources can be provisioned without the need for human interaction from the service provider.
- **Broad network access:** These services are available over the network and can be accessed through standard mechanisms by both thick and thin clients.
- **Resource pooling:** The provider's computing resources are pooled to serve multiple consumers using a multi-tenant model, with different physical and virtual resources dynamically assigned and reassigned according to consumer demand.
- **Rapid elasticity:** Cloud computing services can be rapidly and elastically provisioned to quickly scale out and rapidly released to quickly scale in.
- **Measured service:** Cloud systems automatically control and optimize resource use by leveraging a metering capability at some level of abstraction appropriate to the type of service (e.g., storage, processing, bandwidth, and active user accounts).

III.4.3 Challenges and Drawbacks

Despite its numerous benefits, cloud computing also has potential drawbacks and challenges [46]:

- **Security and Privacy:** When sensitive data is stored on the cloud, it's essential to have faith in the service provider's security measures. However, users generally have little or no control over the tactical security mechanisms used by the provider.
- **Dependency and Vendor Lock-in:** It can be challenging to migrate from one provider to another due to the differences between provider platforms and services.
- **Technical Difficulties and Downtime:** While all major vendors have sophisticated redundancy measures, outages are still a reality in the cloud world. Additionally, users may need to learn new systems and interfaces to effectively use cloud services.

III.4.4 The Role of Cloud Computing in ML

Cloud computing has played a significant role in the democratization and evolution of ML. It has enabled researchers and data scientists to handle the vast computational needs of ML and access state-of-the-art ML tools at affordable prices [112].

III.4.4.1 MLaaSML as a Service

MLaaS is an umbrella definition of automated and semi-automated cloud platforms that cover most infrastructure issues such as data pre-processing, model training, model evaluation, and prediction. Examples of MLaaS include Google Cloud AI, Microsoft Azure ML, and Amazon SageMaker [113].

III.4.4.2 Scalability

Cloud platforms provide the flexibility to scale resources according to the demands of the ML workload. This is especially critical for large-scale ML tasks that require handling vast amounts of data and substantial computational power [113].

III.4.4.3 Speed and Agility

In the rapidly evolving field of ML, the speed and agility offered by cloud computing can provide a significant advantage. Cloud-based models can be deployed quickly, and updates or changes can be made in real-time [113].

III.4.4.4 Data Security and Privacy

While cloud computing offers many advantages, it also presents data security and privacy challenges, particularly in the context of ML. Careful consideration must be given to storing and processing data to ensure compliance with applicable regulations [54].

III.4.5 Examples of Cloud Based Environment

Two notable examples of cloud computing in the field of ML and data science are Google Colab and Microsoft Azure Notebooks. These platforms provide a cloud-based environment for executing and sharing Jupyter notebooks, widely used in data analysis, ML, and related fields.

III.4.5.0.1 Google Colab

Google Colab [114] is a free cloud service offered by Google where you can develop DL applications using popular libraries such as TensorFlow, PyTorch, Keras, and OpenCV. Colab provides GPU and is heavily used for ML, data analysis, and AI research.

III.4.5.0.2 Microsoft Azure Notebooks

Microsoft Azure Notebooks [115] is a similar service provided by Microsoft. In addition to Python, Azure Notebooks also support R and F#, which are popular languages in data science. Azure Notebooks integrate well with various Azure services, which can be beneficial for creating end-to-end data science projects on the cloud.

These platforms highlight the key advantage of cloud computing - the ability to access high-powered computational resources without needing to own expensive hardware or install complex software packages.

III.4.6 Summary and Future Outlook

Cloud computing is indispensable in ML by providing scalability, speed, and accessibility. It offers an agile infrastructure for large-scale ML tasks, allowing researchers and data scientists to experiment with different models and approaches. Despite data security and privacy challenges, cloud computing's advantages remain significant.

Looking towards the future, it is evident that the role of cloud computing in ML will continue to evolve and expand. With technological advancements, we can expect improvements in computation speed, data storage, and security. As we explore innovative ways to handle and process data, cloud computing will undoubtedly continue to shape the landscape of ML.

In the next section, we will examine [The next topic to be discussed], an essential aspect of [context].

III.5 Conclusion

Chapter III furnishes a panoramic exploration of ML, Missing Data, and Cloud Computing. Each of these disciplines plays an indispensable role in our time's rapidly evolving, digital, and data-driven landscape.

ML has catapulted to the forefront as an influential instrument, empowering machines to extrapolate from data and augment their performance iteratively. Nevertheless, the quality and totality of this data can substantially sway the efficacy of ML models, accentuating the importance of proficiently managing missing data.

Simultaneously, Cloud Computing has instigated a radical shift in our data storage, management, and computation methodologies, proffering scalable, accessible, and cost-efficient solutions for deploying complex ML models. Despite potential data security and privacy hurdles, its advantages are transformational and pervasive.

As we look to the future, the confluence of these fields is projected to intensify, propelled by continued technological breakthroughs and an insatiable demand for more intelligent, proficient, and scalable solutions. Consequently, an in-depth understanding of these realms is paramount for anyone navigating the labyrinth of contemporary computing and data science.

Chapter IV

Methodology

IV.1 Introduction

This chapter delves into the research study's in-depth methodological framework, providing a comprehensive overview of the techniques, practices, and decisions made throughout the research process. It offers a deep dive into the research design, data collection, preprocessing methods, and the ML models and their implementation. The study explores two intricate application cases – the detection of BHP flooding attacks on OBS networks and the imputation of missing data in Photovoltaic (PV) system installations. These distinct application cases pose unique challenges requiring the application of intricate ML techniques. This research aims to capitalize on advanced analytics, particularly ML, to develop and validate robust models capable of enhancing the reliability and performance of OBS networks and PV installations.

IV.2 Research Design

IV.2.1 Detailed Account of the Research Design

The research design employs a comprehensive dual-case study format. This structure thoroughly examines the complexity inherent in each application case, focusing on the practical implementation of ML algorithms to address the problem statements. This approach guarantees that ideas from each case are concrete and relevant and significantly contribute to each sector by using real-life examples. Further, it exemplifies how ML models can be adapted and applied in various domains, highlighting their versatility and effectiveness.

IV.2.2 Justification for the Chosen Design

The dual-case study design was adopted due to its alignment with the nature of the problems under investigation. With each case - detection of flooding attacks in OBS networks and imputation of missing data in PV systems - situated in distinct contexts, it was necessary to adopt a design that permitted an exhaustive, nuanced exploration of each problem. The case study design enables this by allowing a focused investigation that can yield insights that a broader approach may miss. Moreover, it ensures scientific rigor and consistency across both studies, reinforcing the validity and reliability of the research outcomes.

IV.3 Data Source and Preprocessing

IV.3.1 Description of Data Sources

The datasets used in the study are derived from real-world scenarios, reinforcing the authenticity and practicality of the research. For the OBS network case, the dataset comprises BHP flooding attack data obtained from the UCI ML Repository. Meteorological data are essential to PV installations were utilized for the PV system case. These datasets reflect real-world scenarios and challenges encountered in each field, thus providing a solid foundation for the study.

IV.3.2 Rationale for Data Source Selection

The selection of datasets was a critical decision driven by the requirement for relevance to the research problems and the potential to generate powerful insights. The BHP flooding attack data accurately mirrors a realistic scenario of network security threats, making it the ideal candidate for exploring novel detection approaches. On the other hand, meteorological data is inherently tied to the performance and efficiency of PV installations, hence, its selection for the PV systems case. The datasets provide a solid foundation for the research and enhance the applicability and generalisability of the study's findings in real-world scenarios.

IV.3.3 Data Preprocessing Techniques

The preprocessing stage is a fundamental part of any ML-based study, involving various steps to prepare the data for subsequent analysis. For the OBS network case, preprocessing included generating a sub-table and selecting relevant parameters to optimize the dataset

for further analysis. In the PV system case, preprocessing involved simulating real-world scenarios of missing data, a common challenge in PV system installations. These steps played a critical role in successfully validating the ML models, ensuring they operate on clean, relevant, and optimally structured data.

IV.4 Implementation of ML Models

IV.4.1 Selection of ML Models

The ML models were chosen based on their alignment with the problem statement, potential performance, and compatibility with the datasets. The Ant Colony Optimization Ant Colony Optimization (ACO) algorithm, SVMs, and Extreme Learning Machines (ELM) were selected for the OBS network case, while SVR was chosen for the PV system case. The selected models' capabilities were tailored to suit the unique demands of each case, ensuring effective and optimal performance.

IV.4.2 Implementation of Models

In implementing these models, attention was paid to each case study's nuances and unique challenges. Modifications were made to existing algorithms to optimize their performance. For instance, in the OBS network case, the ACO algorithm was modified to make it faster and more efficient, with binary information coding and a stopping criterion based on the number of features. Similarly, for the PV system case, existing imputation methods were adjusted and tested for their efficacy in handling missing data, leading to the selection of SVR as the most promising method.

IV.4.3 Model Validation Techniques

Various validation techniques were employed to ensure the robustness and reliability of the ML models. In the OBS network case, the performance of the classifiers was assessed through several simulations, while in the PV system case, the efficiency of the imputation methods was evaluated by comparing their results. These validation techniques confirmed the models' performance and substantiated the study's findings.

IV.5 Conclusion

This chapter has presented the methodological framework guiding this research, including the choice of a dual-case study design, the data sources and preprocessing techniques, and the selection, implementation, and validation of ML models. The methodological depth and breadth have allowed for an exhaustive exploration of the complexities inherent in each case study. From mitigating network security threats in OBS networks to handling missing data in PV installations, the research's methodological rigor ensures these challenges are addressed effectively. The application of advanced ML models in these cases further illustrates the potential of ML in providing innovative solutions to real-world problems. The following chapter, "Application Cases," delves into the intricacies of the implementation and results of each case study in detail, ultimately revealing the power of ML in diverse domains.

Chapter V

Case Applications and Critical Discussion

V.1 Introduction

This comprehensive dissertation strives to elucidate the dynamic intersection between cutting-edge AI and ML methodologies and two critical domains: Network Security and Data Imputation. By illuminating these interdisciplinary arenas, this research underscores the vast potential of sophisticated computational algorithms, mainly when strategically applied to novel and complex challenges. The thesis presents two detailed case studies, each shedding light on a distinct application of these advanced computational techniques.

The inaugural case study navigates the intriguing realm of OBS networks - a revolutionary paradigm in optical communications. The deployment and sustainability of these networks are significantly threatened by BHP flooding attacks. These malicious assaults have the potential to severely impede network functionality, resulting in debilitating Denial of Service (DoS) scenarios. This case study embarks on a journey through a pioneering security approach designed to alleviate this threat. It leverages a unique blend of AI techniques, primarily SVMs and ACO algorithms, to construct an efficacious node classification model adept at detecting and thwarting BHP flooding attacks, thereby safeguarding the seamless operation of OBS networks.

The second case study delves into the ever-important field of renewable energy, focusing on PV systems. The accuracy and completeness of meteorological data significantly influence these systems' performance, efficiency, and reliability. Instances of missing data can substantially deter the system's performance, undermining the efficacy of renewable energy generation. This case study addresses this quandary by proposing a robust data imputation method underpinned by SVR. This study juxtaposes this innovative imputation

technique against traditional approaches, emphasizing its superior capacity for addressing the issue of missing data.

V.2 Case Study 1 - Detection of Flooding Attack on OBS Network

V.2.1 Introduction to the First Study

OBS (in figure V.1) has emerged as a dynamic data switching technique, acting as a synthesis Optical Packet Switching (OPS) and Optical Circuit Switching (OCS). These two methods represent the opposite ends of the spectrum of data switching techniques, with OPS allowing for maximum flexibility and OCS offering the highest efficiency. OBS combines the benefits of both methods, striking a balance between flexibility and efficiency. The technique of OBS initiates with an independent transmission of the header over a reserved optical channel, a step termed as delayed reservation [116]. This process forms the cornerstone of OBS and sets it apart from other data-switching techniques. In essence, the independent header transmission allows for the setup of the optical path before the actual data transmission.

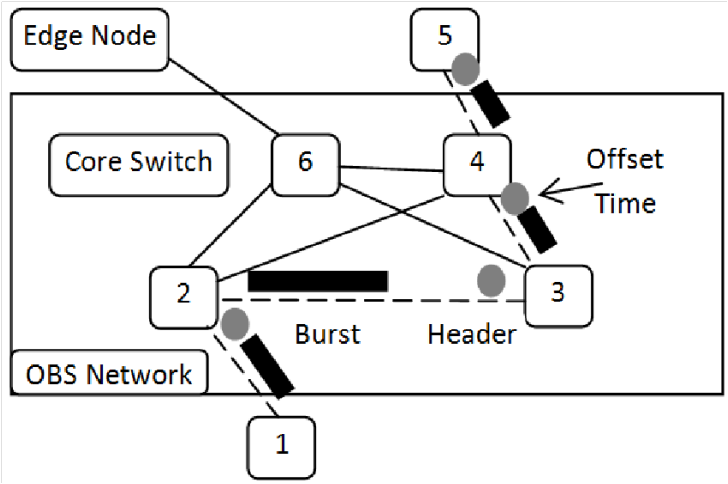


Figure V.1: Optical Infrastructure

The growing reliance on optical media in telecommunications has made OBS a significant player in the sector. With its capability to offer high bandwidth, optical media has become an essential element in telecommunications, thus elevating the relevance of OBS. Importantly, OBS utilizes the high bandwidth offered by optical media and optimizes it. This efficient use of bandwidth, coupled with the elimination of the need for optical memory, has positioned OBS as a significant innovation in data switching techniques.

The OBS technique hinges on data transmission in the form of large packets, also known as bursts. Each of these bursts is preceded by a BHP , which serves as a signaling header. The role of the BHP is crucial to the OBS technique, as it allows for the conversion and processing of the header at each node. This conversion and processing, known as Offset Time (OT) , negates the need for memory usage in the routing of data [117].

Despite the advantages of OBS , it faces a significant threat in the form of flood attacks shown in figure V.2. This kind of attack disrupts the Quality of Service (QoS) offered by OBS . Flood attacks occur when continuous BHPs are sent to a server without the accompanying data, leading to a situation where all Wavelength-Division Multiplexing (WDM) channels are reserved by the emitted BHPs, resulting in DoS [117].

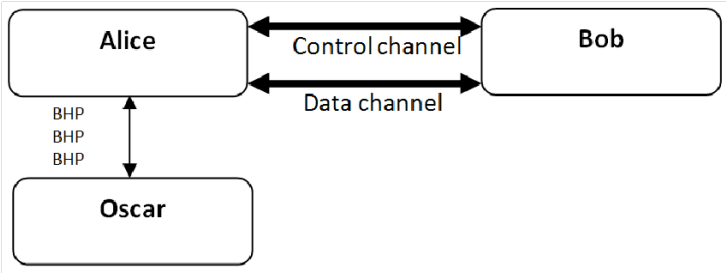


Figure V.2: The Scenario of BHP Flooding Attack

Originally, flood attacks were aimed at Transmission Control Protocol (TCP)/IP network environments. However, they have evolved to target OBS systems, causing significant disruption in the process. In the context of OBS , a flood attack prevents the system from receiving any communication during the attack, as all the channels are occupied, waiting for data that never arrives.

Given the serious implications of flood attacks on OBS , there is a pressing need for effective security measures. To address this, our study aims to propose a novel security approach. This approach is based on a model of node classification that is designed to counter BHP flood attacks. The model utilizes a learning database that comprises four classes. These classes are defined based on several parameters that characterize the states of bandwidth and the types of packets.

V.2.2 Related Works

Several researchers have attempted to address the security challenges faced by OBS networks. These attempts, which are detailed in the literature, offer varying degrees of success.

Rajab et al. [118] proposed a security solution that relies on a decision tree rule learning approach. This approach focuses on counteracting BHP flooding attacks in the

OBS network. The researchers proposed a decision tree-based architecture as a potential solution to the problem. The architecture features a learning algorithm that is capable of extracting rules from tree models. These rules, generated using data processed from several simulation runs, are designed to classify misbehaving edge nodes into four sub-class labels. The researchers found that their method could achieve an accuracy of 87% in classifying the nodes into the four labels, which are MisbehavingBlock (Block), Behaving-No Block (No Block), Misbehaving-No Block (M-No Block), and Misbehaving-Wait (M-Wait).

In a different study, Yayah et al. [119] put forth an Intelligent OT algorithm. This algorithm adapts the OT based on the condition of the network and the traffic. The algorithm's fuzzy input parameters include burst size, destination, and burst queuing delay. The researchers proposed this algorithm as an alternative to adaptive OT algorithms and conventional OT .

Another study by Hasan et al. [120] introduced a Deep Donvolutional Deural Network (DCNN) model for predicting BHP flooding of Optical Burst System. The researchers compared the performance of this model with other ML techniques, such as Naïve Bayes, SVM , and KNN. For the comparison, they employed a comprehensive set of standard performance metrics that are typically used in the design of classification problems. The metrics included Classification Accuracy, Sensitivity, Precision, Specificity, Negative Predictive Value, False Positive Rate, False Negative Rate, F1-Score, and Error Rate of Classification.

Tang et al. [121] proposed a Distributed Denial-of-Service (DDoS) attack assessment method that employs an optimized cloud model. This method assesses DDoS attacks based on the state changes of IP addresses, defining a fusion feature value to establish a V-Support Vector Machines (V-SVM) classification model for network flow analysis.

Finally, Ismail et al. [122] proposed a model for detecting flooding-based DoS attacks in a cloud environment. Their model is structured in three phases, namely, the normal traffic pattern modeling phase, the intrusion detection processes phase, and the prevention phase. The detection is achieved using the covariance matrix mathematical model.

Although these studies provide useful insights into the security problem of OBS networks, they often consider the problem as a simple phenomenon. In reality, however, the problem is complex and dynamic. It is complex because it depends on several factors, and it is dynamic because numerous scenarios may lead to the same result. Our study seeks to address this problem from a different perspective. We propose an approach that combines several techniques to provide enhanced classification precision, parameter optimization, and computation speed.

V.2.3 The Problem of Flooding Attacks in OBS

V.2.3.1 OBS Network

In the contemporary telecommunications landscape, optical mediums have gained significant importance due to their ability to provide high bandwidth. A particular approach, OBS , leverages this high bandwidth and optimizes its usage in a significant manner. One of the main advantages of OBS is its non-reliance on optical memory. The OBS process involves segmenting data into larger packets, otherwise known as 'bursts.' Prior to sending these bursts, OBS reserves the optical channel by dispatching a BHP [123].

This BHP carries essential information, including the OT, which serves as a schedule for data transmission. This time offset allows the conversion and processing of the header at each intermediate node, which mitigates the need for memory usage in data routing.

V.2.3.2 Flooding Attacks

Flooding attacks pose a major threat to network security. The primary outcome of such an attack is DoS , which typically operates within the Transmission Control Protocol TCP framework. The attack consists of sending a series of synchronization requests, or a torrent of BHPs , to the target system.

Initially, an attacker dispatches a large volume of BHPs to the server, reserving the optical channel. Subsequently, the server acknowledges the connection and waits for the incoming data bursts [123]. This waiting duration is predetermined during the server configuration and is dependent on the network latency. However, in a flood attack scenario, the data bursts never arrive. Consequently, the system resources remain in reservation, awaiting data that never materializes. Through repeated iterations of this false alarm, the server becomes incapable of establishing other connections, leading to a DoS state.

OBS has a significant vulnerability to flood attacks due to its channel reservation release mechanism, which operates in both implicit and explicit modes. Implicit release is characterized by the addition of burst length and OT to the BHP . In contrast, explicit release occurs when an end message (REL) is received by the node, which is sent after the data burst. Thus, the continuous transmission of BHPs in a flooding attack leads to network overload and eventual DoS.

V.2.4 Material and Methods

V.2.4.1 Fault Diagnosis

Fault diagnosis is a vital aspect of any system operation, whose primary objective is to discern whether a system is functioning normally or anomalously. In the event of a malfunction, the diagnostic system must accurately identify the affected components, the nature of the malfunction, and the necessary repair actions [124].

For certain complex systems, formulating a typical mathematical model can be challenging. To counter this issue, connection data are employed to construct a classification model. This model is represented as an array of values, each associated with a category identifier.

V.2.4.2 Cloud Computing

Cloud computing has undergone substantial advancements in recent times, attracting considerable investment from numerous companies. This trend has resulted in the creation of various applications, often leading to compatibility issues.

To address these incompatibilities, standardization is the optimal solution. By adhering to existing standards during the development of new applications, compatibility can be ensured. Our application was developed following this principle, adhering to current standards.

The foundation of cloud computing lies in virtualization, enabling developers to create a network and its corresponding virtual machines. However, the creation of numerous disparate platforms can introduce interoperability issues. For instance, if a specific platform is required for our application, the deployment of a new structure can be time-consuming.

To circumvent these challenges, several solutions have been proposed. The Open Virtual Machine Format (OVMF) is a storage standard that supports a multitude of virtualization platforms [125]. It aids in ensuring portability, integrity, and streamlining the installation and configuration phases of virtual machines.

Furthermore, the Open Cloud Computing Interface (OCCI) , developed by the Open Grid Forum Open Grid Forum (OGF) community, provides a suite of protocols and interfaces for deploying applications and managing networks [126].

Another important component in the realm of cloud computing is the Storage Networking Industry Association Cloud Data Management Interface (Advancing Storage and Information Technology Advancing Storage and Information Technology (SNIA) and Cloud Data Management Interface (CDMI)), which handles user interactions with data and implements data encryption.

A cloud computing application is typified by five key properties:

Service availability is dictated by user needs. The system is remotely managed by the user via an intuitive interface. Cloud servers employ a digital transmission network capable of delivering high data rates and offering quick access from any location. Redundancy in data storage across multiple servers allows users to select the server nearest to their location for fast data access. The addition or removal of a machine is facilitated by a simple script, enabling providers to bill customers based on resource usage and duration. Unused processing units do not incur charges. Cloud applications can be categorized into four models:

Private Cloud: Infrastructure is installed, utilized, and managed by a single organization. Community Cloud: Infrastructure is shared among multiple organizations with strong mutual relationships and managed by the organizations themselves or a third party. Public Cloud: Infrastructure is open to the public, with usage billed accordingly. Hybrid Cloud: A combination of the three previous models.

V.2.4.3 Neural Networks

Neural networks have received considerable attention in the literature, particularly regarding their application to fault diagnosis problems. ANNs have been proposed for a multitude of problem areas, including classification and function approximation problems.

Several types of neural networks have been used to address fault diagnosis, with differences primarily in the network's activation function, such as sigmoidal or radial basis, and the learning approach, which may be supervised or unsupervised.

In supervised learning with multilayer neural networks, the problem is reduced to estimating the connection weights. These weight values are calculated by training the network, using the difference between the desired and computed values to guide the search. This approach is suitable for fault diagnosis systems as it can yield accurate classifications.

On the other hand, unsupervised learning employs estimation techniques in a type of neural network known as self-organizing. Here, a set of input observations is repeatedly presented to the input layer of the network, without any direct learning during the self-organization process.

In our study, TensorFlow was utilized, a ML library developed by the Google Brain Team in 2011 [127]. TensorFlow offers a suite of DL functionalities and is open-source, enabling the creation of software components in graphical forms. In these charts, nodes represent mathematical operations, while edges signify the multi-dimensional data arrays communicated between them.

V.2.4.4 SVMs

In our wrapper approach, we employed SVMs as our classifier of choice. Introduced by Vapnik [128], SVM presents an intriguing learning algorithm that exhibits competitive advantages over other ML techniques such as neural networks and decision trees.

Assume a dataset $S = (x, y1), ..., (xi, yi), ..., (xm, ym)$, where $xi \in \mathbb{R}^N$ is a feature vector and $yi \in -1, +1$ is a class label. The goal of the SVM is to find a function of the form:

$$w\phi(x) + b = 0, \quad \text{with} \quad yi(w\phi(xi) + b) \geq 1 - \xi_i \tag{V.1}$$

The function separates the training data set S into two classes (positive and negative), as illustrated in figure V.3. In general, S cannot be partitioned by a linear hyperplane. However, S can be transformed into a higher dimensional feature space to make it linearly separable.

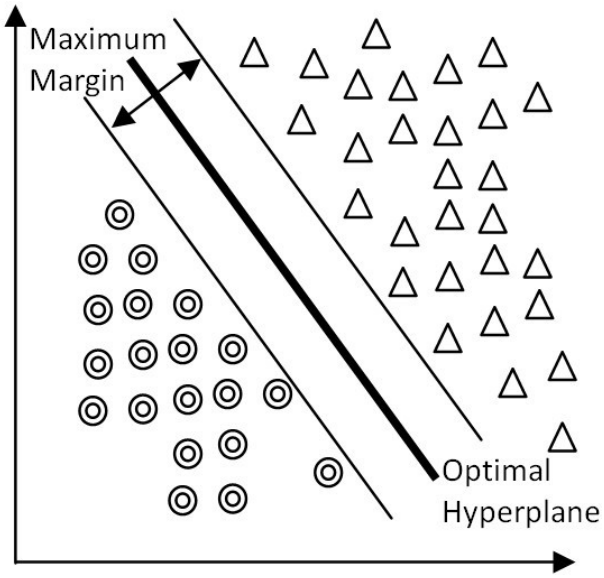


Figure V.3: Linear Two-Classes SVM

An inner product Kernel of the form:

$$K(xi, xj) = \phi(xi).\phi(xj) \tag{V.2}$$

can be used, eliminating the need for explicit computation of the mapping $\phi(x)$. To solve the optimal hyperplane problem, we can construct a Lagrangian and transform it into the dual. We then equivalently maximize:

$$\frac{1}{2}\sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{m}\alpha_i\alpha_j y_i y_j K(x_i, x_j) \tag{V.3}$$

subject to:

$$\sum_{i=1}^{m}\alpha_i y_i = 0, \quad 0 \leq a_i \leq C \tag{V.4}$$

For a test example z, we define the decision function as follows:

$$\text{sign}\left(\sum_{i=1}^{m}\alpha_i y_i K(z_i, z) + b\right) \tag{V.5}$$

Here, w is the weight vector, b is the bias term, C is the punishment parameter, and α is the Lagrange multiplier.

V.2.4.5 ACO

ACO is an algorithm inspired by the cooperative behavior of real ant colonies, known for their ability to find the shortest path from their nest to a food source.

ACO algorithms can be applied to any optimization problem that can be characterized by the following [129]:

- A finite set of components $C = c1, c2, ..., cN$ is given.
- A finite set of L possible connections or transitions among the elements of C is defined over a subset C' of the Cartesian product $C \times C$, $L = c_i c_j | (c_i, c_j) \in C'$, $|L| \leq N^2 c'$.
- For each $l_{c_i c_j} \in L$ a connection cost function $J_{c_i c_j} \equiv J(l_{c_i c_j}, t)$, possibly parameterized by some time measure t, is defined.
- A finite set of constraints $\Omega \equiv \Omega(C, L, t)$ is imposed over the elements of C and L.
- The states of the problem are defined in terms of sequences $s = (c_i, c_j, ..., c_k, ...)$ over the elements of C or L. S' is a subset of S. The elements in S' define the problem's feasible states.
- A neighborhood structure is assigned as follows: the state s_2 is said to be a neighbor of s_1 if s_1 and s_2 are in S and the state S_2 can be reached from s_1 in one logical step, that is, if c_1 is the last component in the sequence determining the state s_1, there must exist $c_2 \in C$ such that $l_{c_1 c_2} \in L$ and $s_2 = \langle s_1, c_2 \rangle$.

- A solution Ψ is an element of S' satisfying all the problem's requirements. A solution is multi-dimensional if it is defined in terms of multiple distinct sequences over the elements of C.
- A cost $J_\Psi(L,t)$ is associated with each solution Ψ. $J_\Psi(L,t)$ is a function of all the costs $J_{c_i c_j}$ of all the connections belonging to the solution.

A distinctive feature of ACO is its probabilistic decision-making approach, which is based on artificial pheromone trails and local heuristic information. This allows ACO to explore a larger number of solutions than greedy heuristics. Another characteristic of the ACO algorithm is the pheromone trail evaporation, a process that gradually decreases the intensity of the pheromone trail over time, aiding in avoiding the rapid convergence of the algorithm towards a sub-optimal region.

In the subsequent section, we present our proposed approach and explain how it is used for the detection of flooding attacks on OBS networks.

V.2.5 System Architecture and Performance Analysis

We will explain in detail the different stages of our approach. Figure V.4 summarizes these steps.

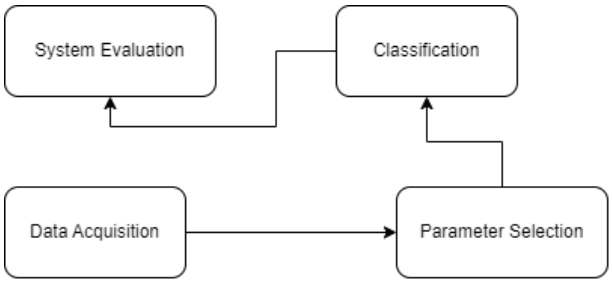


Figure V.4: The Architecture of the Classification Model

Our detection system consists of 5 main steps which are: data acquisition, generation of characteristics, parameter selection, classification, and system evaluation.

The network state is represented by a feature vector x consisting of d parameters, such that $X = (x_1, x_2, \ldots, x_d)$. The vector X represents a point in the representation space. The parameters of the feature vector X are derived from network simulations. When observing new data, the task is to classify it into one of M classes, corresponding to distinct regions in the representation space where similar shapes are grouped together.

We have proposed a set of modifications to the algorithm proposed by Weiqing et al. [130] to make it faster and more efficient. In our detection system, the stopping criterion

is the number of features obtained by the algorithm where the quality of the solution does not improve if we add a new feature or increase the number of iterations.

We have applied a binary coding(in figure V.5), a string of 1s and 0s, to represent the information. The number of parameters is the dimension of the solution.

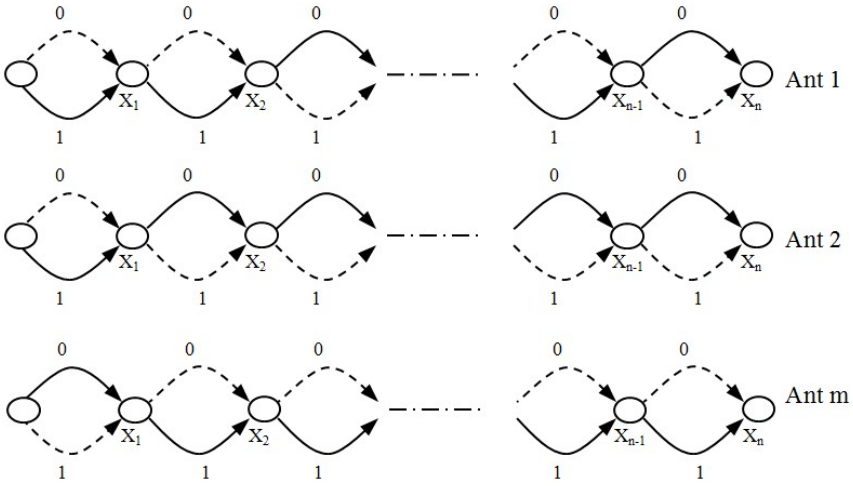


Figure V.5: the Possible Solution Obtained by the ACO Algorithm

The general form of a vector is: $V = (x_1, x_2, \ldots, x_n)$. Where x_i takes the value 1 if the parameter is present in the vector of the solution, otherwise it takes the value 0 if the parameter does not belong to the solution.

Each path of an ant is a solution that consists only of the parameters whose positions equal the value 1:

$$P_m(i,j) = \begin{cases} \frac{\left[\tau_{(i,j)}\right]^{\alpha} \cdot \left[\eta_{(i,j)}\right]^{\beta}}{\sum_{k \in S_m(i)} \left[\tau_{(i,j)}\right]^{\alpha} \cdot \left[\eta_{(i,j)}\right]^{\beta}} & \text{if } j \in S_m(i) \\ 0 & \text{otherwise} \end{cases} \tag{V.6}$$

where:

- $\tau_m(i,j)$: The amount of pheromone deposited by an ant on the arc (i,j).
- α: The pheromone parameter.
- $\eta(i,j)$: The heuristic parameter of the arc (i,j). The heuristic parameter value is fixed and does not change during program execution, is determined by $\eta(i,j) = 1/l_{ij}$, where l_{ij} is the cost for an ant to move between i and j.
- β: The heuristic parameter.
- $S_m(i)$: The set of vertices that remain to be visited by an ant that is on vertex i.

The amount of pheromone is updated using the following rule:

$$\tau(i,j) \leftarrow \rho \cdot \tau(i,j) + \Delta\tau(i,j) \tag{V.7}$$

where:

- ρ: The evaporation parameter.
- $\Delta\tau(i,j)$: The amount of pheromone added.

In our study, we have performed the ACO algorithm using two kinds of classifier SVM and ELM as shown in figure V.6.

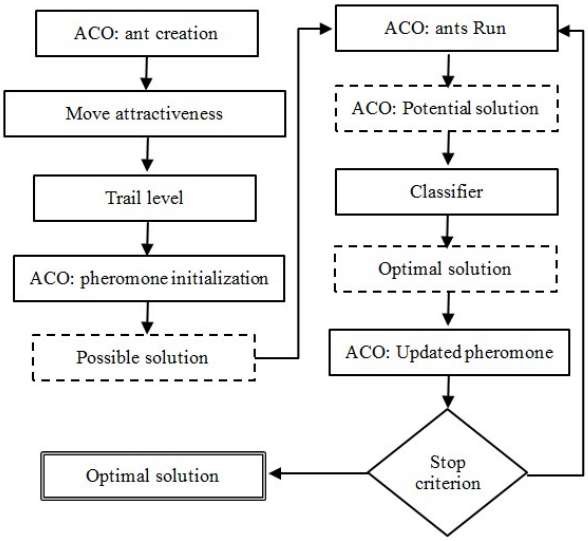


Figure V.6: Optimization of Flooding Attack Detection on OBS Networks

V.2.5.1 Dataset

In this study, we used a BHP flooding attack on an optical burst-switched network data set that is available in the UCI ML repository [131]. This database includes 1075 observations spread over 4 classes which are NB-No Block, Block, No Block, and NBWait. Each observation includes 22 attributes. The database also contains missing values.

V.2.5.2 Features Selection

To generate a sub-table, we performed a preprocessing which is the selection of parameters.

This step consists in carrying out several simulations to fix the parameters of the ACO algorithm and the parameters of the kernel function.

Table V.1: Instances of the obtained database

A3	A4	A17	A18	A19	A20
1000	0	0.7	0	0	B
100	0	0.2	0	0.4	NB
900	0	0.8	0	0	B
100	0	0.3	0	0.4	NB
800	0	0.8	0	0	B
100	0	0.4	0	0.2	NB
700	0	0.8	0	0	B
100	0	0.5	0	0.1	PNB
900	0	0.8	0	0	B
100	0	0.4	0	0.1	PNB

The best pair of (C, γ) is $(2^3, 2^{-5})$. Knowing that we have used the intervals $[2^3, 2^{11}]$, $[2^{-12}, 2^2]$ as search spaces. Table V.1 presents some instances of the obtained database.

The user (abbreviated as 'usr') in the experiments assigns these values (numeric).

- A17. 10-Run-AVG-Drop-Rate: This represents the average packet drop rate over 10 consecutive iterations (numeric).
- A18. 10-Run-AVG-Bandwidth-Use: This denotes the average bandwidth utilized over 10 consecutive iterations (numeric).
- A19. 10-Run-Delay: This refers to the average delay time over 10 consecutive iterations (numeric).
- A20. Node Status' {B, NB, P NB}: This indicates the initial classification of nodes based on Packet Drop Rate, Used_Bandwidth, and Average_Delay_Time_Per_Sec. 'B' signifies Behaving, 'NB' stands for Not Behaving, and 'P NB' denotes Potentially Not Behaving (Categorical).

V.2.5.3 Classification

The original datasets were in ARFF format and required transformation to CSV format. A script was written to perform this conversion and create a CSV file that can be manipulated using Python and Excel.

The data was imported into a Python environment using the pandas library, which allows data manipulation as DataFrames (tables with variable labels as columns and individual labels as lines). These data frames were used to generate visualizations with Matplotlib.

The six classification attributes were extracted using functions from the NumPy library. To visualize the relationships between these attributes, pairwise graphics were created using the Seaborn library, which corrects three deficiencies in Matplotlib illustrated in figure V.7.

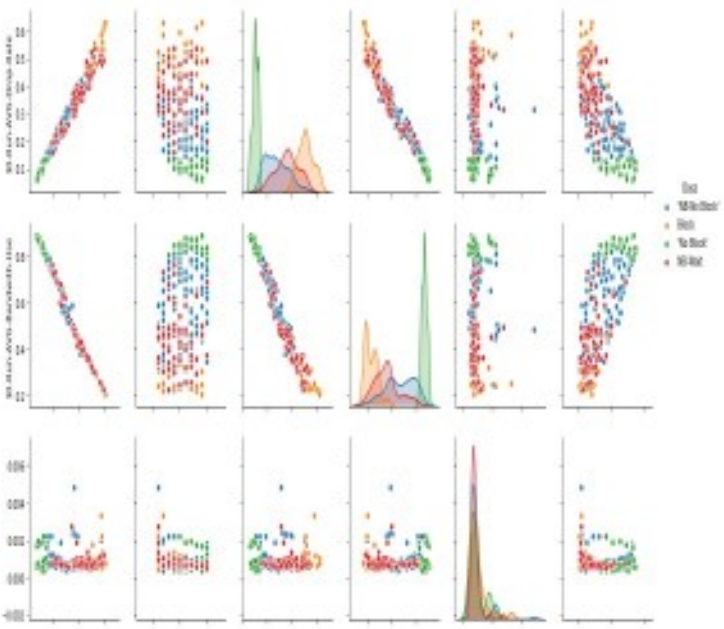


Figure V.7: Correlation Between Attributes

Our model includes a data shuffling step to reduce potential drifts and preserve the model's generality. A portion of the data is kept separate for validation testing. The steps of the preparation process were as follows:

- Assign all columns of the first 8 lines to the variable `data_to_predict`.
- Select the classes of `data_to_predict` and store them in the `predict_class` variable as a NumPy array.
- Assign attributes of `data_to_predict` (excluding classes) to the `prediction` variable.
- Store the remaining variables in `data2`.

In our ongoing work with 'data2', we continue to train and test the neural network. As raw textual data is not directly usable, it necessitates conversion to a digital format, ideally converting each value to a numerical equivalent. This operation in Python is facilitated by the 'LabelEncoder' function of Scikit-learn, a widely-used, open-source ML library.

Our chosen model is a Multilayer Perceptron (MLP), a popular architecture in current applications. Each neuron computes a weighted sum of its inputs and transmits it to a transfer function to produce its outputs. For each layer of the neural network in an MLP , there exists a bias term connected to neurons via a weight, generally referred to as the threshold. Neurons and biases are organized in a feed-forward, non-looping layers structure. Consequently, the MLP can be viewed as an input-output model, with weights and thresholds as the adjustable parameters of the model.

MLPs are capable of modeling complex functions, with the function's complexity determined by the number of layers and units in each layer. Designing MLPs requires specifying the number of hidden layers and the units in these layers. Additionally, selecting suitable activation functions and learning methods is crucial. In our model, we have six input variables (attributes), one output variable with four classes, and six hidden layers, each containing ten neurons.

In Python, we utilize Keras' sequential model. It is essential for this model to be aware of the form of input it will receive, hence, the first layer of a sequential model should be provided with the input shape (later layers will infer it). For multi-class classification problems, the softmax function, a generalization of the sigmoid, serves as an activation function.

Before training, the model requires configuration of the learning process. This is achieved by calling the 'compile' method, which accepts three arguments:

- 'Optimizer': This could either be an optimizer's name (like rmsprop or adagrad) or an instance of the Optimizer class.
- 'Loss': The cost function the model seeks to minimize. It could be referred to by its name.
- 'Metrics': A list of metrics. For classification problems, metrics can be set to ['accuracy']. However, it can be set to another metric if required.

In our model, the loss function is the cross-entropy function, the optimizer is ADAM, and the metric is precision.

We apply the 'fit' function to conduct curve fitting. The essence of curve fitting is to construct a curve from mathematical functions and adjust these functions' parameters to approximate the measured curve. Thus, the term is often used interchangeably with parameter adjustment, curve fitting, profile fitting, or simply fitting.

Regression methods (statistical methods widely used for analyzing the relationship between a variable and one or more others) are applied. In simple cases, it could be multi-linear regression (figure V.8 illustrates the difference between multi and linear Regression)

if the law is linear for all parameters, or polynomial regression if a polynomial simulates the phenomenon. Traditional regression methods allow parameters determination from data calculations but are inapplicable if the function is overly complex. In such cases, a trial-and-error approach, approximating a solution following the least-squares method, is adopted. This solution is not necessarily unique.

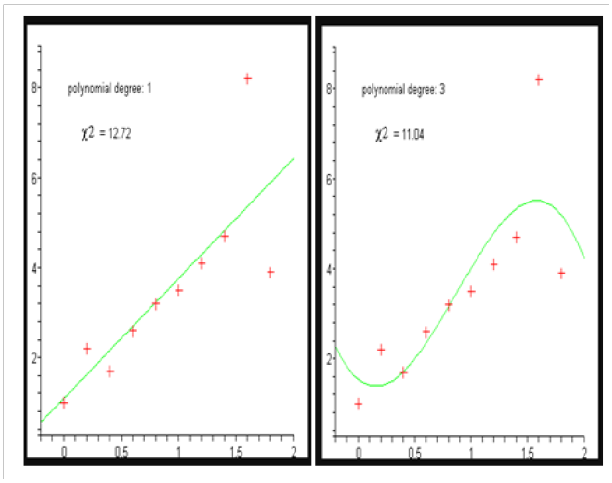


Figure V.8: Multi-Linear and Linear Regression with Fit Function

The `batch_size`, which represents the number of samples used for each learning function update, is set to ten. To associate an evaluation function (named `evaluate`) with our model, we utilize Keras. This function returns the metric (precision or accuracy in our case) and requires the input of attributes and test classes. Additionally, we employ the `predict_classes` function to classify new data, utilizing the two variables we initially reserved (`prediction` and `predict_class`).

Upon examining the results from the SVM and Recurrent Neural Network (RNN) methods, comparing their performance in terms of precision or accuracy, we present the following findings:

For the SVM, we showed the results of two methods: Gaussian Kernel (Accuracy = 91%) and Linear Kernel (Accuracy = 70%). As evident, the Gaussian Kernel outperformed the Linear Kernel in classifying our data. However, the selection of the best kernel varies with each scenario, and it is advised to test all kernels to choose the one that yields the best results with the test data set.

For the RNN, the accuracy was 71%, although this rate could be lower due to the random initialization of weights associated with variables.

In Figure V.9, we present the classification rate obtained using ML algorithms on a randomly generated sub-dataset. Consequently, based on our study, the most effective method to diagnose flooding attacks in OBS is the Gaussian kernel of SVM. This finding is

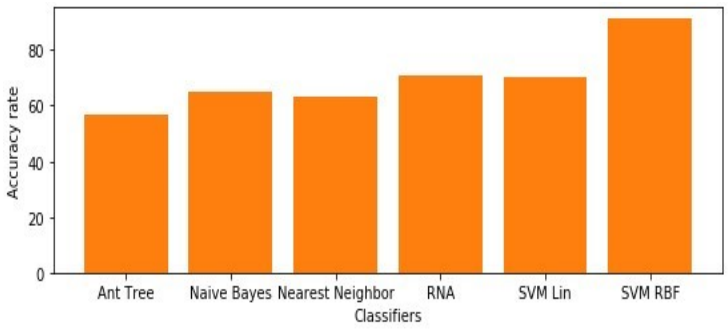


Figure V.9: Detection Accuracy Rate

logical due to the absence of a universally ideal method for determining optimal topology, thereby emphasizing the necessity for advanced research in deep networks.

V.2.6 Case Study 1- Key Findings

We have presented in this paper two methods to detect flooding attacks on the OBS network. We used an ACO algorithm to select a feature subset. This step is mandatory to reduce computing time. It increases the detection accuracy.

We have evaluated the accuracy of our approach using a UCI dataset. Both classifiers SVM and RNA achieve a reasonable detection rate.

Two main routes may be followed in future work. The first is to try another recent approach which is DL . The second is studying the impact of missing data on the detection rate.

V.3 Case Study 2 - Imputation as Service Using Support Vector Regression: Application to a Photovoltaic System in Algeria

V.3.1 Introduction to the Case Study 2

PV systems' installation hinges on two critical elements: physical equipment and meteorological data. The quality of PV system prognosis relies heavily on meteorological data, which can occasionally contain missing values due to various reasons. In spite of the surging volume of data and the advent of Big Data, the problem of missing data continues to persist, especially in the realm of meteorological data. This missing data issue can detrimentally affect the validity and generalization of associations between variables and the internal validity of the data. Ignoring missing data could potentially lead to a significant loss in accuracy. Therefore, the most recommended approach is to impute the missing values [132].

The occurrence of missing values is prevalent in many industries due to various factors such as manual data acquisition, defective sensors or components, and incorrect measurements. Missing data can be categorized into three types: MCAR , MAR , and MNAR . Each type impacts the internal validity of the data in different stages of the research process: sample selection, assignment to the experimental or control group, data collection, and statistical analysis [133].

There are several traditional methods to tackle missing values, such as ignoring the instances with missing data or implementing complete-case or available-case analysis. Moreover, imputation methods like simple imputation, Hot-Deck or Cold-Deck imputation, and MIs have been used extensively. Nevertheless, the issue of missing data in PV systems has not been thoroughly addressed yet, despite the available methodologies [134–136].

A few noteworthy studies attempted to address this gap. For instance, Ioannis and Panapakidis proposed a novel methodology for imputing incomplete data, which uses clustering to group available data patterns into homogeneous clusters [137]. Similarly, Haydar and Zoe have evaluated 36 imputation methods for solar irradiance series over a real dataset recorded in Australia [138]. Furthermore, Tahasin and colleagues have developed an iterative MTL-GP-TS model to predict PV trends by learning/imputing missing values in a time series dataset [139].

V.3.2 PV System

A PV system is a power infrastructure engineered to supply usable solar power through PVs. This system is a conglomeration of solar panels, which absorb and convert sunlight into electricity, a solar inverter to transition from direct current to alternating current, as well as mounting, wiring, and other electrical accessories that make up a working setup (refer to Figure V.10 for essential components). PV systems are occasionally paired with a solar tracking system to enhance performance or integrated with a battery solution [140].

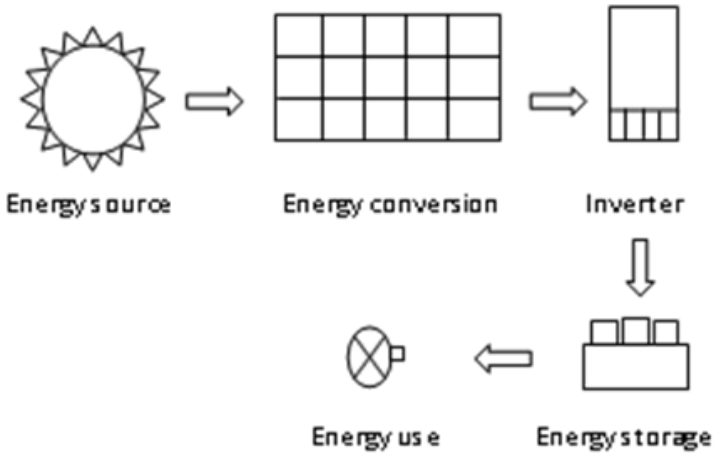


Figure V.10: Components of a PV System.

PV systems come in three categories: residential, commercial, and ground-mounted. The capacity ranges from a few kilowatts for residential units up to hundreds of megawatts for ground-mounted systems. Despite the growing trend towards larger scale systems, rooftop systems continue to dominate the market due to their small size and higher cost per watt [141].

Two primary types of PV systems exist: isolated and grid-connected. Isolated systems, often employed in remote areas without grid access, use batteries for electricity storage and a charge controller for battery longevity [142]. On the other hand, grid-connected systems are directly tied to the grid via an inverter, making it easy for the producer/consumer as the grid balances electricity production and consumption [143].

A PV system is a significant long-term investment, typically boasting a lifespan of more than three decades. Most module manufacturers offer performance guarantees over 20 to 25 years. The lifespan of other components varies; inverters last for 10-15 years, while batteries need replacement roughly every ten years [144–146].

V.3.3 Software Tools

We employ two toolboxes, LIBSVM and PVLIB, using Python to develop our imputation system. LIBSVM, developed by C.Chang in 2001, includes classification and regression approaches, providing a simple interface to utilize different configurations. It comprises two crucial modules: Svmtrain for learning and Svmpredict for classification [147].

PVLIB Python is a comprehensive toolbox offering various functions and classes to model PV systems. Developed from PVLIB MATLAB by Sandia National Laboratories, it serves as a reliable and open platform for PV system modelling. The toolbox provides both procedural code and classes for users preferring object-oriented programming [148].

Our application was implemented on Google Colab, a free cloud-based version of Jupyter Notebook, enabling the use of ML tools in the cloud on virtual machines equipped with TPUs and GPUs.

V.3.4 Proposed Approach

In this section, we address the problem of missing data utilizing several popular imputation methods. These methods generate values to approximate the missing data. We assess the effectiveness of imputation in the context of data classification, predicting class labels. Observations and corresponding labels are divided into a training set and a testing set, the latter serving to estimate the predictive accuracy of the classifier. This research seeks to evaluate the effectiveness of Python's available imputation methods and select the optimal one [149].

Scikit-learn is a Python library dedicated to ML, encompassing functions for estimating random forests, logistic regressions, classification algorithms, and SVMs . It is designed to complement other Python libraries like NumPy and SciPy [150].

The Iris dataset, one of the most well-known in data science, is used extensively in our experiments [151]. This dataset includes 150 equally distributed observations across three species of Iris flowers (Setosa, Versicolor, and Virginica), measuring four characteristics for each observation.

We created artificial datasets containing missing values to apply data imputation methods. For instance, we eliminated 5% of data in a file. We also generated other datasets with varying rates of data deletion and, conversely, datasets with additional data.

Four imputation methods were employed. The first method (M1), Forward Fill (FFill), imputes missing values by filling them with the preceding column or row value. The second method (M2), Backward Fill (BFill), uses the subsequently observed value to fill in the missing value. The third method (M3), Drop, removes records that contain missing values.

In the fourth method (M4), SVR performs Kernel interpolation to fill in the missing value.

Initially, we created a prediction model using the KNN classification method. Table V.2 displays the prediction results for the Iris dataset with imputed copies. According to these results, FFill performed the worst, whereas the other methods achieved similar imputation quality.

Table V.2: Prediction results of Iris dataset.

Data	M1	M2	M3	M4
Original	0.98	0.98	0.98	0.98
MV with 5%	0.97	0.98	0.98	0.98
MV with 10%	0.94	0.97	0.97	0.97

While the Iris dataset is used in this section to demonstrate the impact on classification, in the subsequent section, we use Python's PVLIB to apply our proposed method to our study case of PV systems. PVLIB Python offers functions and classes that simplify obtaining weather forecast data and converting it to PV power. The data from NOAA / NCEP / NWS models can be extracted, including the Global Forecast System (GFS) used in this study [152].

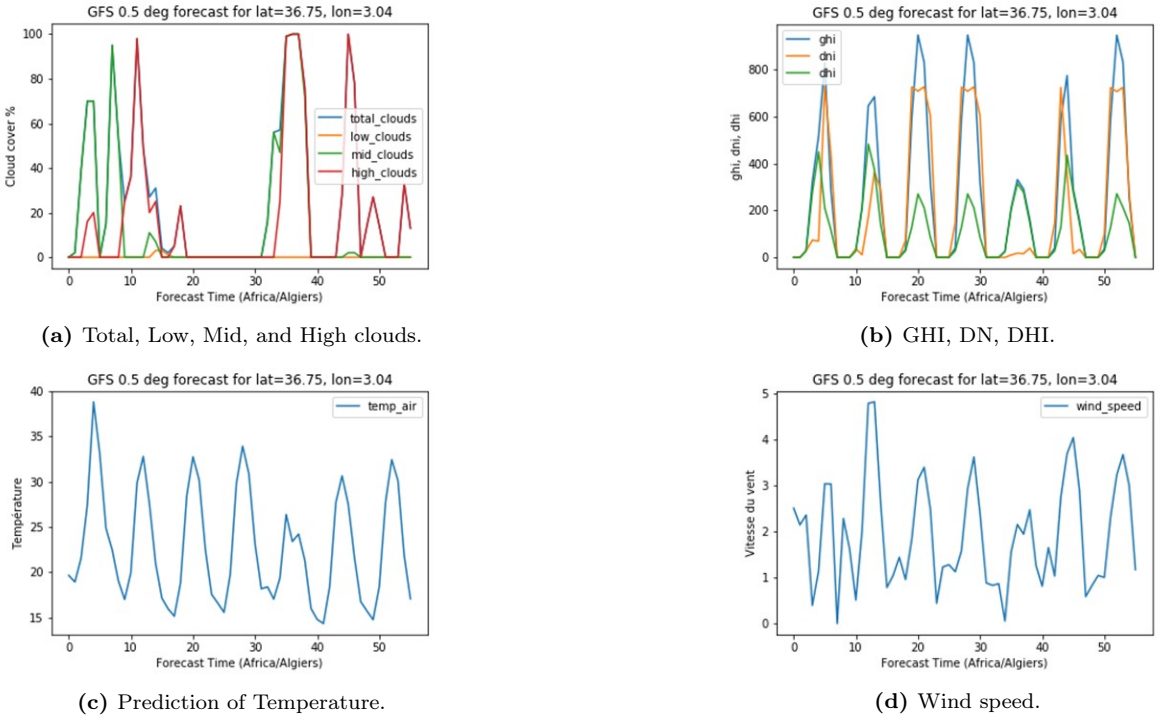
V.3.5 Results

Data obtained using the GFS model follow the American system of units, necessitating modification to match our application's format (see Table V.3). The modified data is saved in CSV format using Pandas to facilitate comparison and evaluate imputation method efficiency.

Table V.3: Example of forecast results on 09-06-2018 (12a.m, 3a.m, and 6a.m).

Parameter	V1	V2	V3
Index2018-06-09	00:00:00	03:00:00	06:00:00
Weather	19.66452	18.928528	21.584686
Wind Speed	2.5121348	2.148635	2.361764
GHI	0	0	29.65
DNI	0	0	28.61
DHI	0	0	27.28
Total Clouds	0	2	39
Low Clouds	0	0	0
Mid Clouds	0	2	39
High Clouds	0	0	0

Figure V.11 display Algeria's forecast results obtained by GFS, showcasing Total, Low, Mid, and High clouds, GHI, DN, DHI, temperature prediction, and wind speed, respectively.



(a) Total, Low, Mid, and High clouds.

(b) GHI, DN, DHI.

(c) Prediction of Temperature.

(d) Wind speed.

Figure V.11: Algeria's Forecast Results Obtained by GFS.

We generated two copies of the origin.csv file to emulate files with missing data. We eliminated 5% in Figure V.12 and 10% of data from the first and second data files, respectively. This operation's result is depicted.

	A	B	C	D	E	F	G	H	I	J
1	index	temp_air	wind_speed	ghi	dni	dhi	total_clouds	low_clouds	mid_clouds	high_clouds
2	2018-06-09 00:00:00+01:00	19.66452	2.5121348	0	0	0	0	0	0	0
3	2018-06-09 03:00:00+01:00	18.928528	2.148635	0	0	0	2	0	2	0
4	2018-06-09 06:00:00+01:00	21.584686	2.361764		28.6123825	27.2856253	39	0	39	0
5	2018-06-09 09:00:00+01:00	27.48233	0.395888	322.143855	73.2091901	275.465732	70	0	70	16
6	2018-06-09 12:00:00+01:00	38.845276	1.14067	516.283999		449.915597	70	0	70	20
7	2018-06-09 15:00:00+01:00	33.09372		828.925278	727.877165	209.859684	0	0	0	0
8	2018-06-09 18:00:00+01:00	24.86676	3.0364952	284.64433	437.49416	116.645714	15	0	15	0
9		22.483612	0.002152221	0	0	0	95	0		0
10	2018-06-10 00:00:00+01:00	19.045074	2.2887406	0	0	0	53	0	53	0
11	2018-06-10 03:00:00+01:00	16.999115	1.6214906	0	0	0	25	0	0	25

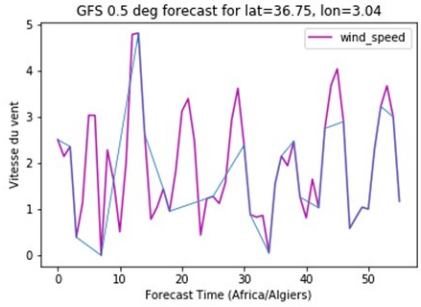
Figure V.12: Data After Deleting 5%.

We applied the imputation as mentioned earlier methods to the Pvlib data, with the code illustrated in Figure V.13.

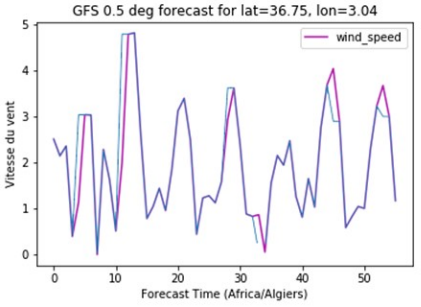
Figures V.14 visually compare the results of wind speed predictions for the original and imputed datasets using different methods. According to these results, FFill is the least effective method, whereas the other techniques demonstrate similar imputation quality. It is important to note that the nature of the data does not significantly influence the imputation quality of methods or the classification or prediction rates.

```
0
import pandas as pd
import matplotlib.pyplot as plt

latitude, longitude, tz = 36.75, 3.04, 'Africa/Algiers'

damaged_data = pd.read_csv("db/damaged_data 5%.csv",sep=';')

imputation_dropna = damaged_data
imputation_ffile  = damaged_data
imputation_bfile  = damaged_data
imputation_intermplation = damaged_data
```

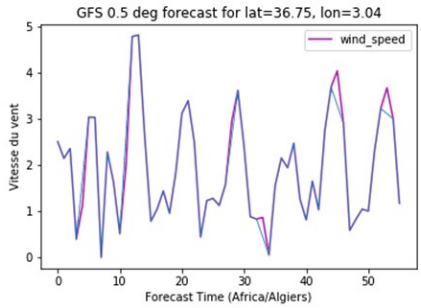
Figure V.13: Dode of Imputation Method.



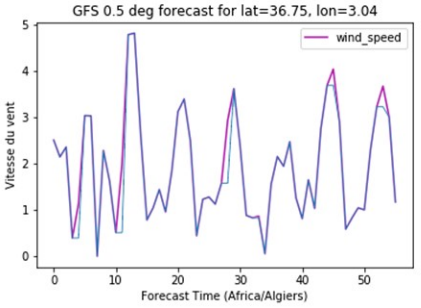
(a) The blue graph describes Pvlib wind speed prediction, and the purple describes the prediction after imputation using fFill.



(b) The blue graph describes Pvlib wind speed prediction, and the purple describes the prediction after imputation using Bfill.



(c) The blue graph describes Pvlib wind speed prediction, and the purple describes the prediction after using Drop.



(d) The blue graph describes Pvlib wind speed prediction, and the purple describes the prediction after imputation using the SVR method.

Figure V.14: Comparison of Wind Speed Prediction Results.

V.3.6 Case study 2- Key Findings

This paper proposes several modules for incomplete data imputation. All PV installations necessitate sensors for physical quantities, which are used to evaluate the system's performance. However, data collected may contain invalid values, including missing ones, leading to potential challenges in system diagnosis. Neglecting observations with missing values can lead to an inaccurate depiction of the system's operation. Hence, it is preferable to impute incomplete data using intelligent techniques. We employed several imputation

methods: FFill, BFill, Drop, and SVR, demonstrating high reliability. This study marks the first step toward developing a comprehensive diagnostic system, which will be the focus of future work.

V.4 Discussion

This section aims to elucidate our research findings and their implications within the broader scope of ML, cybersecurity, and renewable energy management. We'll revisit the two primary studies presented in this thesis. The first case examined the implementation of ML techniques for the detection of BHP flooding attacks in OBS networks, while the second study focused on the application of SVR for the imputation of missing data in PV systems in Algeria.

V.4.1 Interpretation and Implications of the Findings

V.4.1.1 Detection of Flooding Attack on OBS Network

Our research presented a unique methodology to tackle the issue of BHP flooding attacks on OBS networks. By leveraging an ACO algorithm for feature selection and utilizing SVMs and ELM as classifiers, we demonstrated a promising strategy for detecting these complex and detrimental cyber-attacks.

The ACO was inspired by real ant colonies' behavior, exhibiting a probabilistic decision-making process that considered artificial pheromone trails and local heuristic information. This strategy allowed the algorithm to explore more solutions than traditional methods. Our choice of SVM and ELM was driven by their superior effectiveness in partitioning data into classes and learning directly from the data. Combining these methodologies led to high detection rates, enhancing the security measures for OBS networks.

This finding extends the applications of ML in cybersecurity and paves the way for further research in this direction. It addresses a persistent challenge in telecommunication networks, enhancing their resilience to cyber-attacks, which is paramount in our increasingly digital world.

V.4.1.2 Imputation as Service Using SVR

Our second case study addressed a recurring issue in renewable energy management—handling missing data in PV systems. We successfully applied SVR as an imputation method, demonstrating its efficiency in dealing with missing meteorological data, crucial for optimal PV installation operation in Algeria.

The SVR technique capitalizes on the principles of SVM and extends its application to regression problems, providing a powerful tool to handle missing data. We compared the efficiency of SVR with several other imputation methods and found it to be an effective solution for maintaining the reliability of PV installations.

Our study has relevant implications for renewable energy management. It highlights the potential of ML applications in tackling complex issues in this field, underpinning the effective operation and management of PV systems, and contributing to sustainable development goals.

V.4.2 Limitations of the Study

While our research yielded encouraging results, some limitations need to be acknowledged. Our first study on detecting flooding attacks in OBS networks was conducted on a specific dataset from the UCI ML repository. While our method demonstrated high accuracy in this context, its applicability might vary across different datasets and real-world applications, warranting further validation.

Similarly, our second study on data imputation in PV systems was conducted in the context of a specific installation in Algeria. While our method was effective in this setting, the performance might vary across different geographical locations, PV system designs, and with variations in environmental conditions.

V.4.3 Recommendations for Future Work

Our research points towards several potential directions for future work. Regarding the detection of flooding attacks, it would be worthwhile to explore the potential of DL techniques to improve our proposed methods' generalizability. This could enhance the applicability of our approach across diverse datasets and different cybersecurity threats.

Regarding the PV system data imputation, it would be interesting to examine the efficiency of other ML algorithms, perhaps even hybrid approaches, to improve the accuracy of results across various settings. Future studies could also delve deeper into understanding the impact of missing data on PV installations' overall performance and efficiency.

V.5 Conclusion

The compelling case studies presented in this dissertation underscore the transformative and versatile nature of AI and ML techniques when astutely applied to complex real-world dilemmas. The first case study demonstrates how a harmonious combination of SVMs

and ACO algorithms can be harnessed to formulate a robust and proactive security model equipped to mitigate BHP flooding attacks in OBS networks. This study offers a prototype for developing enhanced security protocols, ensuring these networks' secure and robust functioning in the face of ever-evolving cyber threats.

The second case study reveals the superiority of a SVR -based approach for handling missing data in PV systems. This sophisticated imputation technique outperforms traditional methodologies in terms of efficiency and performance, providing an invaluable tool for bolstering the performance of renewable energy systems.

While the insights from these case studies contribute significantly to their respective domains, it is critical to underscore the necessity for perpetual innovation and research. As the wheel of technological advancement continuously turns, it unfurls a plethora of new challenges that demand innovative solutions. Therefore, this research's methodologies and strategies should be considered foundational stepping stones for future research rather than ultimate solutions. Future research directions could encompass the integration of advanced DL techniques, a detailed investigation of the impact of missing data on detection rates, and the development of a comprehensive diagnostic system for PV installations. This perpetual spirit of inquiry will continue to drive progress and innovation in these fascinating and impactful areas.

Chapter VI

Conclusion

VI.1 Recap of the Research

VI.1.1 Initial Problem Statement and Research Objectives

We started our research journey by recognizing two significant issues: the susceptibility of Optical Burst Switching (OBS) networks to Burst Header Packet (BHP) flooding attacks and the handling of missing data in Photovoltaic (PV) installations. Our initial objectives were to devise practical solutions to these problems by leveraging the power of Machine Learning (ML) algorithms.

VI.1.2 Summary of the Research Process

Our research process encompassed formulating relevant research questions, conducting a comprehensive literature review, developing and testing suitable methodologies, and interpreting the results. Specifically, we employed a range of ML models to address the issue of BHP flooding attacks in OBS networks and deployed Support Vector Regression (SVR) to tackle missing data in PV installations.

VI.1.3 Key Findings of the Research

Our findings indicated that ML models could considerably bolster network security by precisely detecting and mitigating BHP flooding attacks. We established that SVR-based data imputation for PV installations was superior to traditional techniques in managing missing data, leading to optimized PV system performance.

VI.1.4 Interpretation of the Results

The results validated our initial hypothesis that ML algorithms could offer promising solutions for OBS network security and handling missing data in PV installations. Moreover, the findings underlined the transformative potential of ML in addressing a wide array of complex real-world challenges.

VI.2 Contribution to Knowledge

VI.2.1 Discussion on Contribution to Knowledge

This research significantly contributes to several fields, including network security, renewable energy, and machine learning applications. We demonstrated how ML models could be adapted to enhance OBS network security and how SVR could be harnessed for improved data imputation in PV systems.

VI.2.2 Implications of the Findings

Our research has vast implications for the broader scientific and technological communities. In network security, particularly within OBS networks, introducing ML models for threat detection and mitigation sets the stage for an improved and more resilient network infrastructure. Regarding renewable energy management, the novel application of SVR for data imputation opens up possibilities for increased efficiency and reliability of renewable energy systems, especially PV installations.

VI.2.3 Potential Impact on Industry Practices

Our research outcomes could lead to a rethinking of current industry practices. Enterprises could implement our proposed ML-based solutions in network security to better protect their network systems. The renewable energy sector could also benefit by adopting our SVR-based approach for managing missing data, enhancing the efficiency of PV systems, and improving overall energy output.

VI.3 Recommendations for Future Research

VI.3.1 Suggestions for Future Research

Several intriguing avenues for future research have emerged from this study. Exploring ML models to counter various types of network attacks could provide a broader spectrum of protection for network security. Also, developing adaptive ML models that can evolve in response to changing threat scenarios is another promising line of inquiry. In the field of renewable energy, future research could aim to enhance the SVR-based data imputation method or examine the potential of other ML techniques, such as predictive analytics, to forecast energy generation more accurately.

VI.3.2 Limitations and Future Challenges

While our research has made significant strides, we acknowledge certain limitations. The effectiveness of ML models in detecting network attacks is subject to the quality of the training data, and as such, the collection and preparation of high-quality data sets should be a focal point for future studies. In the context of PV installations, the performance of the SVR-based imputation method is contingent on the inherent characteristics of the missing data. Further exploration of the most effective ways to handle different types of missing data is warranted.

VI.3.3 Final Reflections on the Research Process

Reflecting on the research journey, it has been both demanding and fulfilling. We were required to gain an in-depth understanding of ML techniques, network security mechanisms, and the intricacies of renewable energy systems. Despite the challenges, our achievements testify to the benefits of a systematic and thorough research approach. They also emphasize the potential of ML as a tool for solving complex issues, thereby offering promising prospects for future scientific exploration and innovation.

Bibliography

[1] T. Mitchell. *Machine Learning.* McGraw Hill, 1997.

[2] Seung Seog Han, Moon Soo Kim, Wonjoon Lim, Gyeong Hun Park, Inho Park, Sung Eun Chang, and Wonwoo Lee. Classification of the clinical images for benign and malignant cutaneous tumors using a deep learning algorithm. *Journal of the American Academy of Dermatology*, 78(6):1106–1109, 2018.

[3] Ryan Poplin, Avinava V Varadarajan, Karl Blumer, Yifan Liu, Michael V McConnell, and Greg S Corrado. Prediction of cardiovascular risk factors from retinal fundus photographs via deep learning. *Nature Biomedical Engineering*, 2(3):158–164, 2018.

[4] P. Mell and T. Grance. The nist definition of cloud computing. *NIST Special Publication*, 800(145):1–7, 2011.

[5] G. Litjens, T. Kooi, B. E. Bejnordi, A. A. A. Setio, F. Ciompi, M. Ghafoorian, J. A. van der Laak, B. van Ginneken, and C. I. Sánchez. A survey on deep learning in medical image analysis. *Medical image analysis*, 42:60–88, 2017.

[6] Z. Zheng, M. R. Lyu, and G. Sun. Cloud-based software engineering: A systematic mapping study. *IEEE Transactions on Services Computing*, 9(6):906–919, 2015.

[7] Andre Esteva, Brett Kuprel, Roberto A Novoa, Justin Ko, Susan M Swetter, Helen M Blau, and Sebastian Thrun. Dermatologist-level classification of skin cancer with deep neural networks. *Nature*, 542(7639):115–118, 2017.

[8] Varun Gulshan, Lily Peng, Marc Coram, Martin C Stumpe, Derek Wu, Arunachalam Narayanaswamy, et al. Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs. *JAMA*, 316(22):2402–2410, 2016.

[9] Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE transactions on image processing*, 26(7):3142–3155, 2017.

[10] Robert Sipos, Dmitry Fradkin, Fabian Moerchen, and Zhuang Wang. Log-based predictive maintenance. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1867–1876. ACM, 2014.

[11] Anna L Buczak and Erhan Guven. A survey of data mining and machine learning methods for cybersecurity intrusion detection. *IEEE Communications Surveys & Tutorials*, 18(2):1153–1176, 2016.

[12] Mohamed Takieddine Seddik, Kadri Ouahab, Chakir Bouarouguene, and Houssem Brahimi. Detection of flooding attack on obs network using ant colony optimization and machine learning. *Computación y Sistemas*, 25, 2021.

[13] Donald B Rubin. Inference and missing data. In *Biometrika*, pages 581–592. JSTOR, 1976.

[14] Zhengping Che, Sanjay Purushotham, Kyunghyun Cho, David Sontag, and Yan Liu. Recurrent neural networks for multivariate time series with missing values. *Scientific reports*, 8(1):6085, 2018.

[15] D. K. Lim, N. U. Rashid, J. B. Oliva, and J. G. Ibrahim. Unsupervised imputation of non-ignorably missing data using importance-weighted autoencoders. *arXiv preprint arXiv:2101.07357*, 2021.

[16] Xiaoqing Liu, Kunlun Gao, Bo Liu, Chengwei Pan, Kongming Liang, Lifeng Yan, Jiechao Ma, Fujin He, Shu Zhang, Siyuan Pan, et al. Advances in deep learning-based medical image analysis. *Health Data Science*, 2021, 2021.

[17] Sotirios Bakas, Mauricio Reyes, Andras Jakab, Stefan Bauer, Markus Rempfler, Alessandro Crimi, Russell T Shinohara, et al. Identifying the best machine learning algorithms for brain tumor segmentation, progression assessment, and overall survival prediction in the brats challenge. In *arXiv preprint arXiv:1811.02629*, 2018.

[18] Alvin Rajkomar, Jeffrey Dean, and Isaac Kohane. Machine learning in medicine. *New England Journal of Medicine*, 380(14):1347–1358, 2019.

[19] Z. M. Çınar, A. A. Nuhu, Q. Zeeshan, O. Korhan, M. Asmael, and B. Safaei. Machine learning in predictive maintenance towards sustainable smart manufacturing in industry 4.0. *Sustainability*, 12(19):8211, 2020.

[20] S. Strecker, W. Van Haaften, and R. Dave. A modern analysis of aging machine learning based iot cybersecurity methods. *arXiv preprint arXiv:2110.07832*, 2021.

[21] Jihoon Moon, Nicolas Hernandez Minaya, Nhat Anh Le, Hyo Choi Park, and Sungyoung Choi. Can ensemble deep learning identify people by their gait using data collected from multi-modal sensors in their insole? *Sensors*, 20(14):4001, 2020.

[22] L. Fremond, V. H. Koelzer, N. Horeweg, and T. Bosse. The evolving role of morphology in endometrial cancer diagnostics: From histopathology and molecular testing towards integrative data analysis by deep learning. *Frontiers in Oncology*, 12:928977, 2022.

[23] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[24] E. Christodoulou, J. Ma, G. S. Collins, E. W. Steyerberg, J. Y. Verbakel, and B. Van Calster. A systematic review shows no performance benefit of machine learning over logistic regression for clinical prediction models. *Journal of Clinical Epidemiology*, 110:12–22, 2019.

[25] G. James, D. Witten, T. Hastie, and R. Tibshirani. *An Introduction to Statistical Learning*, volume 112. Springer, 2013.

[26] S. R. Safavian and D. Landgrebe. A survey of decision tree classifier methodology. *IEEE transactions on systems, man, and cybernetics*, 21(3):660–674, 1991.

[27] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[28] M. Chen, Y. Hao, K. Hwang, L. Wang, and L. Wang. Disease prediction by machine learning over big data from healthcare communities. *IEEE Access*, 5:8869–8879, 2017.

[29] A. Natekin and A. Knoll. Gradient boosting machines, a tutorial. *Frontiers in neurorobotics*, 7:21, 2013.

[30] T. Chen and C. Guestrin. Xgboost: A scalable tree boosting system. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 785–794, 2016.

[31] C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[32] R. Burbidge, M. Trotter, B. Buxton, and S. Holden. Drug design by machine learning: support vector machines for pharmaceutical data analysis. *Computers & Chemistry*, 26(1):5–14, 2001.

[33] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

[34] A. Rajkomar, E. Oren, K. Chen, A. M. Dai, N. Hajaj, M. Hardt, P. J. Liu, X. Liu, J. Marcus, M. Sun, P. Sundberg, H. Yee, K. Zhang, Y. Zhang, G. Flores, G. E. Duggan, J. Irvine, Q. Le, K. Litsch, A. Mossin, J. Tansuwan, D. Wang, J. Wexler, J. Wilson, D. Ludwig, S. L. Volchenboum, K. Chou, M. Pearson, S. Madabushi, N. H. Shah, A. J. Butte, M. D. Howell, C. Cui, G. S. Corrado, and J. Dean. Scalable and accurate deep learning with electronic health records. *NPJ Digital Medicine*, 1(1):18, 2018.

[35] J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA., 1967.

[36] Anil K Jain. Data clustering: 50 years beyond k-means. *Pattern recognition letters*, 31(8):651–666, 2010.

[37] I. T. Jolliffe and J. Cadima. Principal component analysis. *Springer*, 2002.

[38] M. Ringner. What is principal component analysis? *Nature biotechnology*, 26(3):303–304, 2008.

[39] Xiao Wen, Yuanchang Xie, Liming Jiang, Ziyuan Pu, and Tingjian Ge. Applications of machine learning methods in traffic crash severity modelling: current status and future directions. *Transport Reviews*, 41:855 – 879, 2021.

[40] Prasanalakshmi Balaji, Salem Alelyani, Ayman Qahmash, and Mohamed Mohana. Contributions of machine learning models towards student academic performance prediction: A systematic review. *Applied Sciences*, 2021.

[41] Nathan Lau, Lex Fridman, Brett J. Borghetti, and John D. Lee. Machine learning and human factors: Status, applications, and future directions. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 62:135 – 138, 2018.

[42] Sabidur Rahman, Nilton F. S. Seixas, Mahmuda Naznin, and Gustavo Bittencourt Figueiredo. Automation of photonic networks using machine learning: Case studies and future works. *IEEE Photonics Technology Letters*, 33:1317–1320, 2021.

[43] Naila Habib Khan and Awais Adnan. Urdu optical character recognition systems: Present contributions and future directions. *IEEE Access*, 6:46019–46046, 2018.

[44] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash. Edge computing for the internet of things: A case study. *IEEE Internet of Things Journal*, 6(2):2270–2283, 2019.

[45] Tanja Tornede, Alexander Tornede, Jonas Hanselle, Marcel Wever, Felix Mohr, and Eyke Hüllermeier. Towards green automated machine learning: Status quo and future directions. *arXiv preprint arXiv:2111.05850*, 2021.

[46] I. A. T. Hashem, I. Yaqoob, N. B. Anuar, S. Mokhtar, A. Gani, and S. U. Khan. The rise of "big data" on cloud computing: Review and open research issues. *Information Systems*, 47:98–115, 2015.

[47] A. Botta, W. De Donato, V. Persico, and A. Pescapé. Integration of cloud computing and internet of things: A survey. *Future Generation Computer Systems*, 56:684–700, 2016.

[48] D. Li, K. Ota, and M. Dong. Deep learning for smart industry: Efficient manufacture inspection system with fog computing. *IEEE Transactions on Industrial Informatics*, 14(10):4665–4673, 2018.

[49] P. D. Haghighi, S. Zeadally, and X. He. Demystifying cloud-based machine learning services. *IT Professional*, 19(3):38–45, 2017.

[50] H. Choudhury, J. Sengupta, V. Leung, J. Loo, and A. Vinel. Secure and privacy-preserving machine learning for iot data analytics: A survey. *IEEE Communications Surveys & Tutorials*, 22(4):2317–2351, 2020.

[51] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[52] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. *N/A*, 2017.

[53] T. Chen, M. Li, Y. Li, M. Lin, N. Wang, M. Wang, T. Xiao, B. Xu, C. Zhang, and Z. Zhang. Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems. *arXiv preprint arXiv:1512.01274*, 2015.

[54] D. A. B. Fernandes, L. F. B. Soares, J. V. Gomes, M. M. Freire, and P. R. M. Inácio. Security issues in cloud environments: a survey. *International Journal of Information Security*, 13(2):113–170, 2014.

[55] L. Wang, J. Tao, M. Kunze, A. C. Castellanos, D. Kramer, and W. Karl. The architecture of cloud computing. *IEEE International Conference on High Performance Computing and Communications*, 2010.

[56] Donald B Rubin. Inference and missing data. *Biometrika*, 63(3):581–592, 1976.

[57] Zhongheng Zhang. Missing data imputation: focusing on single imputation. *Annals of translational medicine*, 4(1), 2016.

[58] Julie Scheffer. Dealing with missing data. *Research letters in the information and mathematical sciences*, 3(1):153–160, 2002.

[59] R.J.A. Little and D.B. Rubin. *Statistical Analysis with Missing Data*. Wiley Series in Probability and Statistics. Wiley, 2019.

[60] Gustavo EAPA Batista and Maria Carolina Monard. A study of k-nearest neighbour as an imputation method. *His*, 87:48–48, 2002.

[61] Donald B Rubin. *Multiple imputation for nonresponse in surveys*, volume 81. John Wiley & Sons, 2004.

[62] Jinsung Yoon, James Jordon, and Mihaela Schaar. Gain: Missing data imputation using generative adversarial nets. *arXiv preprint arXiv:1806.02920*, 2018.

[63] Christoph Molnar. *Interpretable machine learning*. Lulu. com, 2020.

[64] Guo Haixiang, Yijing Li, Shang Jennifer, Mingyun Gu, Yuanyue Huang, and Bing Gong. Learning from imbalanced data: open challenges and future directions. *Progress in Artificial Intelligence*, 5(4):221–232, 2017.

[65] Cynthia Dwork and Aaron Roth. Algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407, 2014.

[66] Amit Verma and Shivani Kaushal. Deadline-aware scheduling for big data processing in cloud. In *2015 International Conference on Green Computing and Internet of Things (ICGCIoT)*, pages 1625–1629. IEEE, 2015.

[67] Tom M. Mitchell. *Machine Learning*. McGraw-Hill, New York, 1997.

[68] Google Developers. Google machine learning glossary. https://developers.google.com/machine-learning/glossary, n.d. Accessed 2nd March 2020.

[69] Andriy Burkov. *The hundred-page machine learning book*, volume 1. Andriy Burkov Canada, 2019.

[70] Ron Kohavi et al. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Ijcai*, volume 14, pages 1137–1145. Montreal, Canada, 1995.

[71] Andrew Ng. Machine learning yearning. *URL: http://www. mlyearning. org/(96)*, 139, 2017.

[72] J. Bell. *Machine Learning: Hands-On for Developers and Technical Professionals*. Wiley, 2020.

[73] A. Géron. *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. O'Reilly Media, 2019.

[74] C.C. Aggarwal. *Data Classification: Algorithms and Applications*. Chapman & Hall/CRC Data Mining and Knowledge Discovery Series. Taylor & Francis, 2014.

[75] A.J. Izenman. *Modern Multivariate Statistical Techniques: Regression, Classification, and Manifold Learning*. Springer New York, 2013.

[76] John A Hartigan and Manchek A Wong. Algorithm as 136: A k-means clustering algorithm. *Journal of the royal statistical society. series c (applied statistics)*, 28(1):100–108, 1979.

[77] O. Chapelle, B. Schölkopf, and A. Zien. *Semi-supervised Learning*. MIT Press, 2010.

[78] X. Zhu and A.B. Goldberg. *Introduction to Semi-supervised Learning*. Synthesis lectures on artificial intelligence and machine learning. Morgan & Claypool, 2009.

[79] Xiaojin Jerry Zhu. Semi-supervised learning literature survey. *University of Wisconsin-Madison Department of Computer Sciences*, 2005.

[80] R.S. Sutton and A.G. Barto. *Reinforcement Learning, second edition: An Introduction*. Adaptive Computation and Machine Learning series. MIT Press, 2018.

[81] C. Szepesvári. *Algorithms for Reinforcement Learning*. Synthesis lectures on artificial intelligence and machine learning. Morgan & Claypool, 2010.

[82] N. Matloff. *Statistical Regression and Classification: From Linear Models to Machine Learning*. CRC Press, 2017.

[83] C.M. Bishop. *Pattern Recognition and Machine Learning*. Springer New York, 2016.

[84] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016.

[85] Evelyn Fix and Joseph Lawson Hodges. Discriminatory analysis, nonparametric estimation: consistency properties. *Report 4, Project nº 21-49*, 4, 1951.

[86] Bernhard E Boser, Isabelle M Guyon, and Vladimir N Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152, 1992.

[87] Nello Cristianini and Elisa Ricci. Support vector machines. In *Encyclopedia of Algorithms*, pages 928–932. Springer-Verlag, 2008.

[88] David M Schnyer, Peter C Clasen, Christopher Gonzalez, and Christopher G Beevers. Evaluating the diagnostic utility of applying a machine learning algorithm to diffusion tensor mri measures in individuals with major depressive disorder. *Psychiatry Research: Neuroimaging*, 264:1–9, 2017.

[89] S. Rasoul Safavian and David Landgrebe. A survey of decision tree classifier methodology. *IEEE transactions on systems, man, and cybernetics*, 21(3):660–674, 1991.

[90] J. Ross Quinlan. C4. 5: programs for machine learning. *Elsevier*, 2014.

[91] Thomas G Dietterich. Overfitting and undercomputing in machine learning. *ACM computing surveys (CSUR)*, 27(3):326–327, 1995.

[92] Leo Breiman, Jerome Friedman, Charles J Stone, and Richard A Olshen. *Classification and regression trees*. CRC press, 1984.

[93] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[94] Simon Haykin. *Neural networks and learning machines*, volume 3. Pearson Upper Saddle River, NJ, USA:, 2009.

[95] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

[96] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. In *Parallel distributed processing: explorations in the microstructure of cognition*, volume 1, pages 318–362. MIT Press, 1986.

[97] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[98] Finale Doshi-Velez and Been Kim. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*, 2017.

[99] Craig K Enders. *Applied missing data analysis*. Guilford press, 2010.

[100] Aleksey Bilogur. Missingno: a missing data visualization suite. *Journal of Open Source Software*, 3(22):547, 2018.

[101] John W Graham, Scott M Hofer, and David P MacKinnon. Maximizing the usefulness of data obtained with planned missing value patterns: An application of maximum likelihood procedures. *Multivariate behavioral research*, 31(2):197–218, 1996.

[102] Donald B Rubin. Inference and missing data. *Biometrika*, 63(3):581–592, 1976.

[103] Donald B Rubin. *Multiple imputation for nonresponse in surveys*. John Wiley & Sons, 1987.

[104] James L Peugh and Craig K Enders. A practical guide to multilevel modeling. *Journal of School Psychology*, 48(1):85–112, 2010.

[105] Joseph L Schafer and John W Graham. Missing data: our view of the state of the art. *Psychological methods*, 7(2):147, 2002.

[106] Olga Troyanskaya, Michael Cantor, Gavin Sherlock, Patrick Brown, Trevor Hastie, Robert Tibshirani, David Botstein, and Russ B Altman. Missing value estimation methods for dna microarrays. *Bioinformatics*, 17(6):520–525, 2001.

[107] Daniel J Stekhoven and Peter Bühlmann. Missforest—non-parametric missing value imputation for mixed-type data. *Bioinformatics*, 28(1):112–118, 2012.

[108] Lovedeep Gondara. Multiple imputation for categorical data using autoencoders. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 1184–1190. IEEE, 2018.

[109] Alex J Smola and Bernhard Schölkopf. A tutorial on support vector regression. *Statistics and computing*, 14(3):199–222, 2004.

[110] Toby Velte, Anthony Velte, and Robert Elsenpeter. *Cloud computing: a practical approach*. McGraw-Hill, Inc., 2010.

[111] Judith Hurwitz, Robin Bloor, Marcia Kaufman, and Fern Halper. *Cloud computing for dummies*. Wiley Publishing, 2010.

[112] Zhen Zhang, Cheng Zhang, and Guan Zhou. Cloud computing technologies for connected government. *Connected Government: Concepts and Applications*, pages 18–31, 2016.

[113] Xing Liu, Jian Shen, Yanzhi Chen, Qi Quan, Hu Yang, and Hong Song. A survey on security issues in services communication of microservices-enabled fog applications. *Concurrency and Computation: Practice and Experience*, 2017.

[114] Ekaba Bisong. *Building Machine Learning and Deep Learning Models on Google Cloud Platform: A Comprehensive Guide for Beginners*. Apress, 2019.

[115] Gopika Premsankar, Lobna Ismail, Tarik Taleb, and Konstantinos Samdanis. Container orchestration for iot: Kubernetes vs docker swarm. *2018 IEEE Global Communications Conference (GLOBECOM)*, pages 1–7, 2018.

[116] H.S. Saini and A. Wason. Fallacious node algorithm for performance enhancement in opticalburst-switching networks. *Journal of Optical Communications*, 40(3):239–245, 2019.

[117] A. Rajab, C.T. Huang, M. Al-Shargabi, and J. Cobb. Countering burst header packet flooding attack in optical burst switching network. In *International Conference on Information Security Practice and Experience*, pages 315–329, 2016.

[118] A. Rajab, C.T. Huang, and M. Al-Shargabi. Decision tree rule learning approach to counter burst header packet flooding attack in optical burst switching network. *Optical Switching and Networking*, 29:15–26, 2018.

[119] Y. Rahmat-Samii, V. Manohar, J.M. Kovitz, R.E. Hodges, G. Freebury, and E. Peral. Development of highly constrained 1 m ka-band mesh deployable offset reflector antenna for next generation cubesat radars. *IEEE Transactions on Antennas and Propagation*, 67(10):6254–6266, 2019.

[120] R. Hasan, R. Sion, and M. Winslett. Preventing history forgery with secure provenance. *ACM Transactions on Storage (TOS)*, 5(4):1–43, 2009.

[121] Dan Tang and Xiaohong Kuang. Distributed denial of service attacks and defense mechanisms. In *IOP Conference Series: Materials Science and Engineering*, volume 612, page 052046. IOP Publishing, 2019.

[122] M.N. Ismail, A. Aborujilah, S. Musa, and A. Shahzad. Detecting flooding based dos attack in cloud computing environment using covariance matrix approach. In *Proceedings of the 7th international conference on ubiquitous information management and communication*, pages 1–16, 2013.

[123] M.K.H. Patwary and M.M. Haque. A semi-supervised approach to detect malicious nodes in obs network dataset using gaussian mixture model. In *Inventive Communication and Computational Technologies*, pages 707–719, 2020.

[124] O. Kadri, L.H. Mouss, and M.D. Mouss. Fault diagnosis of rotary kiln using svm and binary aco. *Journal of Mechanical Science and Technology*, 26(2):601–608, 2012.

[125] J. Matthews, T. Garfinkel, C. Hoff, and J. Wheeler. Virtual machine contracts for datacenter and cloud computing environments. In *Proceedings of the 1st Workshop on Automated Control for Datacenters and Clouds*, pages 25–30, 2009.

[126] M.R. Nelson. Building an open cloud. *Science*, 324(5935):1656–1657, 2009.

[127] Martín Abadi. Tensorflow: learning functions at scale. In *Proceedings of the 21st ACM SIGPLAN International Conference on Functional Programming*, pages 1–1, 2016.

[128] V. Vapnik. *The Nature of Statistical Learning Theory*. Information Science and Statistics. Springer New York, 2013.

[129] W. Xiong, L. Wang, and C. Yan. Binary ant colony evolutionary algorithm. *International Journal of Information Technology*, 12(3):10–20, 2006.

[130] Quamar Niyaz, Weiqing Sun, and Ahmad Y Javaid. A deep learning based ddos detection system in software-defined networking (sdn). *arXiv preprint arXiv:1611.07400*, 2016.

[131] Adel Rajab. Burst Header Packet (BHP) flooding attack on Optical Burst Switching (OBS) Network. UCI Machine Learning Repository, 2017. DOI: https://doi.org/10.24432/C51C81.

[132] Ouahab Kadri, LH Mouss, and Adel Abdelhadi. Fault diagnosis for a milk pasteurisation plant with missing data. *International Journal of Quality Engineering and Technology*, 6(3):123–136, 2017.

[133] Yuqing Zhang, Akram Alyass, Thuva Vanniyasingam, Behnam Sadeghirad, Iván D Flórez, Sathish Chandra Pichika, Sean Alexander Kennedy, Ulviya Abdulkarimova, Yuan Zhang, and Tzvia Iljon. A systematic survey of the methods literature on the reporting quality and optimal methods of handling participants with missing outcome data for continuous outcomes in randomized controlled trials. *Journal of Clinical Epidemiology*, 88:67–80, 2017.

[134] Gleb Beliakov, Daniel Gómez, Simon James, Javier Montero, and J Tinguaro Rodríguez. Approaches to learning strictly-stable weights for data with missing values. *Fuzzy Sets and Systems*, 325:97–113, 2017.

[135] Zhongheng Zhang. Missing data imputation: focusing on single imputation. *Annals of Translational Medicine*, 4(1), 2016.

[136] Matthew Blackwell, James Honaker, and Gary King. A unified approach to measurement error and missing data: overview and applications. *Sociological Methods and Research*, 46(3):303–341, 2017.

[137] Ioannis P Panapakidis and Athanasios S Dagoumas. Day-ahead electricity price forecasting via the application of artificial neural network based models. *Applied Energy*, 172:132–151, 2016.

[138] Haydar Demirhan and Zoe Renwick. Missing value imputation for short to mid-term horizontal solar irradiance data. *Applied Energy*, 225:998–1012, 2018.

[139] Tahasin Shireen, Chenhui Shao, Hui Wang, Jingjing Li, Xi Zhang, and Mingyang Li. Iterative multi-task learning for time-series modeling of solar panel pv outputs. *Applied Energy*, 212:654–662, 2018.

[140] F Baumgartner. *Photovoltaic (PV) balance of system components: Basics, performance*, pages 135–181. Elsevier, 2017.

[141] Peter O Ogbeche and Lawrence O Ohiero. Empirical estimation of monthly average daily solar radiation and solar electricity output from sunshine hours in ogoja in nigeria. *International Journal of Innovative Research and Development*, 7(7):5, 2018.

[142] Eniko Lazar, Dorin Petreus, Radu Etz, and Toma Patarau. Software solution for a renewable energy microgrid emulator. *Advances in Electrical and Engineering, Computer*, 18(1):89–94, 2018.

[143] Makbul AM Ramli, Ssennoga Twaha, Kashif Ishaque, and Yusuf A Al-Turki. A review on maximum power point tracking for photovoltaic systems with and without shading conditions. *Renewable and Sustainable Energy Reviews*, 67:144–159, 2017.

[144] Heverton A Pereira, Francisco D Freijedo, MM Silva, VF Mendes, and R Teodorescu. Harmonic current prediction by impedance modeling of grid-tied inverters: A 1.4 mw pv plant case study. *International Journal of Electrical Power and Energy Systems*, 93:30–38, 2017.

[145] Fred Ka-Wai Kong, Man-Chung Tang, Yi-Chun Wong, Maggie Ng, Mei-Yee Chan, and Vivian Wing-Wah Yam. Strategy for the realization of efficient solution-processable phosphorescent organic light-emitting devices: design and synthesis of bipolar alkynylplatinum (ii) complexes. *Journal of the American Chemical Society*, 139(18):6351–6362, 2017.

[146] Daniel Akinyele, Juri Belikov, and Yoash Levron. Battery storage technologies for electrical applications: Impact in stand-alone photovoltaic systems. *Energies*, 10(11):1760, 2017.

[147] Guido Van Rossum, Fred L Drake, et al. *Python reference manual*. Centrum voor Wiskunde en Informatica Amsterdam, 1995.

[148] William F Holmgren, Robert W Andrews, Antonio T Lorenzo, and Joshua S Stein. Pvlib python 2015. In *2015 ieee 42nd photovoltaic specialist conference (pvsc)*, pages 1–5. IEEE, 2015.

[149] Wes McKinney et al. Data structures for statistical computing in python. In *Proceedings of the 9th Python in Science Conference*, volume 445, pages 51–56. Austin, TX, 2010.

[150] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, and Vincent Dubourg. Scikit-learn: Machine learning in python. *The Journal of Machine Learning Research*, 12:2825–2830, 2011.

[151] Jeremy N Bailenson, Emmanuel D Pontikakis, Iris B Mauss, James J Gross, Maria E Jabon, Cendri AC Hutcherson, Clifford Nass, and Oliver John. Real-time classifi-

cation of evoked emotions using facial feature tracking and physiological responses. *International journal of human-computer studies*, 66(5):303–317, 2008.

[152] G Gutman and A Ignatov. The derivation of the green vegetation fraction from noaa/avhrr data for use in numerical weather prediction models. *International Journal of remote sensing*, 19(8):1533–1543, 1998.